МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра економічної кібернетики

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
економічної кібернетики
_____ Іванченко Н.О.
«___»_____ 2020 р.

КВАЛІФІКАЦІЙНА РОБОТА

(Пояснювальна записка)

Здобувача освітнього ступеня «Магістр»

Тема: Розробка бази даних для організації обліку комплектуючих на підприємстві

Виконала: Бахорчук Вікторія Олександрівна

Керівник: д.т.н., професор Олешко Т.І.

Консультанти з розділів:

Розділ 1: д.т.н., професор Олешко Т.І.

Розділ 2: д.т.н., професор Олешко Т.І.

Розділ 3: д.т.н., професор Олешко Т.І.

Нормоконтролер із ЄСКД (ЄСПД):

к.е.н., доцент Густера О.М.

Київ-2020

Національний авіаційний університет
Факультет економіки та бізнес-адміністрування
Кафедра економічної кібернетики
Освітній ступінь «Магістр»
Напрям підготовки «Цифрова економіка»

ЗАТВЕРДЖУЮ
Завідувач кафедри
економічної кібернетики
_____Іванченко Н.О.
«____» _____2020 р.

ЗАВДАННЯ НА ДИПЛОМНУ РОБОТУ

Студентки: Бахорчук Вікторії Олександрівни

Тема роботи: «Створення бази даних для обліку організації комплектуючих на підприємстві»

затверджена наказом ректора № 1967/ст. від 08.10.2020 р.

1. Термін здачі студентом закінченої роботи на кафедру "17" грудня 2020 р.

2. Вихідні дані до роботи: використано матеріали конференцій та інтернет-ресурси, а також періодична література.

3. Зміст дослідження (перелік питань до розробки):

- охарактеризовано теоретичні засади баз даних та систем управління базою даних;
- досліджено етапи проектування бази даних та їх особливості;
- розроблено реляційну модель бази даних предметної області;
- оптимізовано продуктивність обліку книг в БД бібліотеки «Світогляд»;
- проведено технічний аналіз рівня якості варіантів реалізації функцій;
- проведено економічний аналіз та обрано кращий варіант програмного продукту техніко-економічного рівня.

4. Перелік обов'язкових демонстраційних матеріалів (до захисту магістерської роботи): 10 слайдів.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Позначки керівника про виконання завдань
1	Отримання завдання на кваліфікаційну роботу	05.10	
2	Огляд літератури за темою	06.10	
3	Проаналізовано класифікацію баз даних	10.09	
4	Охарактеризовано реляційну модель	20.09	
5	Розглянуті етапи проектування БД	30.09	
6	Побудова реляційної моделі предметної області	15.10	
7	Оптимізація продуктивності СУБД	21.10	
8	Проведено функціонально-вартісного аналізу програмного продукту	15.11	
9	Аналіз отриманих результатів	20.11	
10	Створення слайдів та написання доповіді	30.11	
11	Попередній захист випускної роботи	14.12	
12	Корегування роботи за результатами попереднього захисту	14.12	
13	Остаточне оформлення кваліфікаційної роботи та слайдів	15.12	
14	Підписання відгуку та рецензії	18.12	
15	Захист кваліфікаційної роботи у ДЕК	21.12	

5. Дата видачі завдання “ 05 ” 10 2020 р.

Керівник _____ **д.т.н., професор Т.І. Олешко**

(підпис)

Завдання прийняв для виконання _____ В.О. Бахорчук

(підпис)

РЕФЕРАТ

Бахорчук Вікторія Олександівна. Створення бази даних для організації обліку комплектуючих на підприємстві. – Кваліфікаційна робота магістра зі спеціальності 051 «Економіка», ОПП «Цифрова економіка». Національний авіаційний університет Міністерства освіти та науки України, Київ, 2020.

У даній роботі був проведений комплексний аналіз баз даних та реляційної моделі предметної області, було побудовано базу даних для бібліотеки «Світогляд», було проведено функціонально-вартісний аналіз програмного продукту.

Дипломна робота складається зі вступу, трьох розділів, висновків, переліку джерел посилання.

Дипломна робота містить 103 сторінки, 45 ілюстрацій, 15 таблиць та 53 джерел літератури.

Метою дослідження є створення и даних для організації обліку книжкового фонду бібліотеки «Світогляд» за допомогою структурної мови програмування SQL.

Об'єктом дослідження є процес розробки бази даних для реалізації обліку книжкового фонду бібліотеки «Світогляд».

Предметом дослідження є реляційна модель бази даних для організації обліку книг для бібліотеки «Світогляд».

Наукова новизна дослідження полягає у створенні бази даних для бібліотеки «Світогляд», що обумовлено на сьогоднішній день необхідністю використання великих, і постійно зростаючих, обєягів інформації при вирішенні статистичних, управлінських та інших завдань.

Ключові слова: *база даних, система управління базами даних, реляційна модель, інфологічний та даталогічний рівні проектування.*

ЗМІСТ

ВСТУП

Основні ідеї сучасних інформаційних технологій базуються на концепції, згідно з якою дані повинні бути організовані в базу даних з метою адекватного відображення мінливого реального світу і задоволення інформаційних потреб користувачів. Такі бази даних створюються і функціонують під управлінням спеціальних програмних комплексів, які називаються системами управління баз даних (СУБД).

Одним з ключових напрямків в області автоматизація бізнес-процесів з використанням інформаційних технологій є розробка баз даних, що дозволяють розв'язати проблему зберігання і систематизації інформації згідно з індивідуальними вимогами компанії.

Збільшення обсягу та структурної складності збережених даних, розширення кола користувачів інформаційних систем призвели до широкого поширення найбільш зручних і порівняно простих для розуміння реляційних (табличних) СУБД. Для забезпечення одночасного доступу до даних безліні користувачів, нерідко розташованих досить далеко один від одного і від місця зберігання баз даних, створені мережні мультикористувацькі версії БД заснованих на реляційній структурі. У них тим або іншим шляхом вирішуються специфічні проблеми паралельних процесів, цілісності (правильності) і безпеки даних, а також санкціонування доступу.

Основою для обліку, контролю та планування на підприємстві слугують різні типи картотек, реєстраційних журналів та списків, так як з часом вони періодично наповнюються та обновляються. При великому обсязі даних пошук та узагальнення необхідних відомостей, що здійснюються ручним способом, являють собою достатньо трудомісткий процес. З цієї причини і з'явилась потреба у створенні, формуванні та реалізації бази даних.

Без баз даних на сьогодні неможливо уявити роботу більшості фінансових, промислових, торговельних та інших організацій. Вони

дозволяють структурувати, зберігати, обновляти та виймати інформацію оптимальним для користувача способом.

База даних – це організована згідно з визначеними правилами та підтримувана в пам'яті комп'ютера сукупність даних, що характеризується актуальним станом деякої предметної області та використовується для задоволення інформаційних потреб користувачів.

Стрімкий розвиток інформаційних технологій призводить до їх впровадження в багатьох сферах діяльності людини. Винятком не стала і бібліотечна система. Поточний етап розвитку бібліотеки в цілому ставить перед собою таке завдання як вдосконалення діяльності бібліотеки. Автоматизовані бібліотечні системи являють собою такі системи, які дозволяють планувати ресурси організацій для бібліотек, за допомогою яких здійснюється відстеження бібліотечних фондів, що контролюють всі етапи.

Розробка БД в магістерській роботі виконується для певної предметної області, в нашому випадку це бібліотека «Світогляд». З перших кроків приходиться враховувати всі її особливості, проводиться вивчення предметної області та розробляється її формалізований опис. На основі отриманих результатів, проводиться логічне проектування. Після чого здійснюється розробка запитів, форм та звітів. База даних дозволяє своєчасно редагувати новий потік даних та отримати достовірну та в повному обсязі інформацію в пошуку об'єкта, який необхідно виділити з великих масивів даних.

Дана база даних буде створюватись для керівництва бібліотеки. Їм у свою чергу завжди потрібно мати оперативний доступ до інформації. Для цього необхідна загальна база даних, яка включає всю необхідну інформацію. Універсальність бази даних визначена можливістю її постійного заповнення новою інформацією, до того ж, у великих об'ємах по мірі зростання надходження книг. Таким чином, створення бази даних, яка володіє такими особливостями, являється дуже корисною та застосування її в бібліотеці «Світогляд» вважається необхідним завданням, яке і буде виконано в магістерській роботі.

Існує безліч систем управління базами даних (СУБД), призначених для різних платформ. СУБД відіграють важливу роль в більшості організацій. Різниця між цими системами доволі велика, однак їх об'єднує лише одне: всі вони підтримують SQL (мова структурованих запитів), як засіб доступу до даних та управління ними. За допомогою SQL можна створювати реляційні бази даних та вилучати з них корисну інформацію. Дана мова програмування прийнята в якості міжнародного стандарту, а також повсюдно застосовується для створення та обслуговування реляційних баз даних.

Актуальність роботи: створення бази даних для бібліотеки «Світогляд» обумовлена сьогодні необхідністю використання великих, і постійно зростаючих обсягів інформації при вирішенні статистичних, управлінських та інших завдань. У зв'язку зі збільшенням обсягу книжкового фонду, а також для поліпшення контролю проведення, обумовлює актуальність бази даних для бібліотеки «Світогляд». Актуальність розробки даної бази даних для предметної області бібліотеки «Світогляд» продиктована необхідністю повної автоматизації всіх процесів роботи працівників даної установи, що в результаті дозволить значно оптимізувати процес роботи даної організації і відповідно знизити трудовитрати співробітників.

Мета дослідження: створення бази даних для організації обліку книжкового фонду бібліотеки «Світогляд» за допомогою структурної мови програмування SQL.

Для досягнення мети магістерської роботи необхідно виконати перелік таких поставлених завдань:

1. Проаналізувати сучасні технології створення БД і СУБД.
2. Охарактеризувати етапи проектування бази даних та їх особливості.
3. Розробити реляційну модель бази даних для організації обліку книг бібліотеки «Світогляд» за допомогою мови програмування Transact-SQL.

4. Оптимізувати продуктивність обліку книг в БД бібліотеки «Світогляд».
5. Провести технічний аналіз рівня якості варіантів реалізації функцій.
6. Провести дослідження програмного продукту з економічної точки зору та порівняти запропоновані варіанти реалізації функцій.

Постановка задачі – необхідно створити інформаційну базу даних у сфері Microsoft SQL Server. Вибір інструмента проектування обумовлений тим, що СУБД Microsoft SQL Server широко доступний в вільному доступі в мережі інтернет.

Для прийняття обґрунтованих та ефективних рішень у виробничій діяльності, в управлінні економікою та в політиці сучасного спеціаліста необхідно вміти за допомогою комп'ютерів та засобів зв'язку отримувати, накопичувати, зберігати та обробляти дані, представляючи результат у вигляді наглядових документів. Тому тема цієї роботи є актуальною. Таке навчання, як бібліотека, незалежно, потребує комп'ютеризації своєї діяльності та переходу від роботи з паперовими носіями до роботи з електронними носіями. Ведення видачі клієнтам книг, що знаходяться в книгосховищі, за допомогою записів у журналах і картотеках, виживає себе. Тому поставлена задача про необхідність розробки баз даних для бібліотеки, за допомогою якої співробітники бібліотеки зможуть частково (а в майбутньому повністю) відмовитись від ведення записів на папері і, яка допоможе автоматизувати деякі процеси за відсутності навчальної книги в бібліотеці (наприклад, зберігати в базі даних у вигляді таблиці якісь значення, що повторюються, які при ручному веденні в обліку доводились би багаторазово переписувати ще раз).

Об’єкт дослідження – процес розробки бази даних для реалізації обліку книжкового фонду бібліотеки «Світогляд».

Предмет дослідження – реляційна модель бази даних для організації обліку книг для бібліотеки «Світогляд».

РОЗДІЛ 1

ТЕОРЕТИЧНІ ЗАСАДИ ОСНОВ БАЗ ДАНИХ

1.1. Бази даних та системи управління базами даних

За три останніх десятиліття стало загальновизнаним, що інформація є не менш важливим ресурсом людського суспільства, ніж сировина, енергія і їжа. Можна стверджувати, що практично в будь-якому вигляді людської діяльності необхідно задоволення інформаційних потреб тою чи іншою мірою. Так, наприклад, збираючись на вулицю ми завжди хочемо отримати інформацію про погоду. Більшість з нас в тому чи іншому вигляді щодня отримують різну інформацію з газет, радіо, телепередач, Інтернету. Не кажучи вже про джерела професійної інформації.

Розвиток систем зв'язку і комунікацій призвело до ускладнення і диференціації інформаційних процесів в людському суспільстві. Здатність накопичувати інформацію і забезпечувати ефективний доступ до неї стає визначальним фактором не тільки розвитку людського суспільства, а й підтримки його життєздатності. Швидке зростання обсягів інформації, закріпленої на зовнішніх стосовно до людини носіях, привів до появи нових суспільних інститутів (бібліотеки, архіви, преса, обчислювальних центрів і т. д.) і спеціальних систем (служби науково-технічної інформації, довідкові служби, глобальні інформаційні комп'ютерні мережі).

Розвиток засобів обчислювальної техніки та інформаційних технологій відкрило нові можливості і способи зберігання, уявлення і пошуку інформації, зокрема, створення обчислювальних систем, "доступних на вимогу" - тобто обчислювальні ресурси стають таким же доступним ресурсом для споживання людиною, як електроенергія, природний газ, вода.

Таким чином, різко зростають вимоги до якості і надійності проектування систем для роботи з інформацією, що надається в електронному вигляді. Слово "інформація" пов'язане з широким колом понять. Інформація – відомості про що-небудь, незалежно від форми їх подання. З точки зору як користувачів, так і розробників ІС, у інформації є одна важлива властивість – вона є одиницею даних, що підлягають обробці. Зазвичай інформація надходить споживачеві саме у вигляді даних: таблиць, графіків, малюнків, фільмів, усних повідомлень, які фіксують в собі інформацію певної структури і типу. Таким чином, дані виступають як спосіб представлення інформації в певній, фіксованій формі, придатній для обробки, зберігання та передачі [10].

Дані (data) – піддається різній інтерпретації представлення інформації у формалізованому вигляді, придатному для передачі, зв'язку, або обробки. Ключових моментів тут два. По-перше, інформація повинна бути представлена у формалізованому вигляді (іншими словами, підкорятися певним правилам). По-друге, різна інтерпретація дозволяє нам по-різному сприймати одні й ті ж дані в різному контексті [1, с.12].

Вимоги до організації даних інформаційних систем:

1) Інтеграція даних – коли всі дані зберігаються централізовано, створюючи динамічно обновлювану модель.

2) Максимальна незалежність прикладних програм від даних або забезпечення фізичної та логічної незалежності даних.

Виконання цих вимог призвело до створення єдиного для всіх завдань блоку даних – бази даних і розробки однієї керуючої програми для маніпулювання даними на фізичному рівні – СУБД (рис. 1.1).

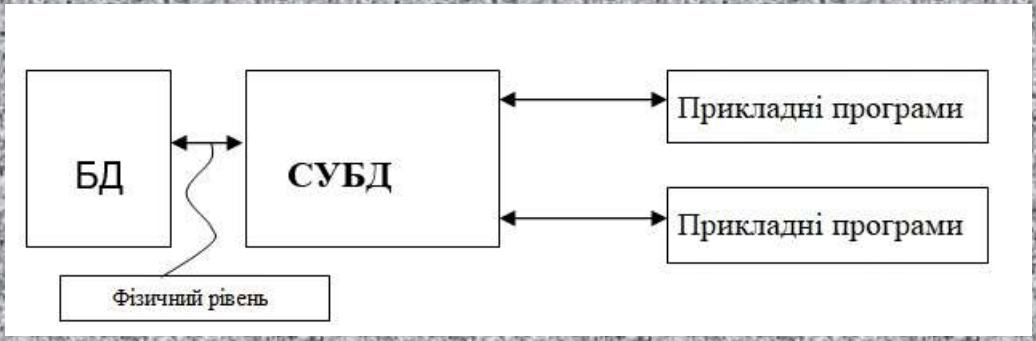


Рис. 1.1. Блок даних

Саме СУБД забезпечує незалежність даних, зміна фізичної організації сприймається СУБД і не впливає на прикладну програму. З іншого боку, зміна логіки програми не вимагає реорганізації і зміни механізму доступу до фізичних даних. Введення СУБД поділяє логічну структуру даних від фізичної структури даних. Відмінною рисою сучасних БД слід вважати спільне зберігання даних з їх описом. Сучасний підхід вимагає, щоб в програмі були задані лише імена і формати даних, що обробляються. Поставляючи дані в програму, СУБД їх попередньо обробляє, у зв'язку з чим зміна організації даних не відбивається на прикладних програмах, в цьому випадку міняються тільки процедури СУБД. Більшість автоматизованих систем тим чи іншим чином використовують структуровану інформацію. У сучасних додатках такі упорядковані дані заведено зберігати в базах даних – особливих файлах, використання яких разом зі спеціальними програмними засобами дозволяє користувачеві, як переглядати необхідну інформацію, так і маніпулювати нею. База даних (database) – сукупність даних, організованих відповідно до концептуальної структури, яка описує характеристики цих даних і взаємини між відповідними сутностями і підтримуючої одну або більше областей застосування [2].

Поняття база даних увійшло в широкий ужиток досить-таки пізно і при цьому втратила більшу частину свого первинного сенсу. Для когось база даних є довільним набором інформаційних елементів. Інші ж визначають це поняття більш строго.

Коротко сформулюємо основні сучасні принципи організації баз даних:

- значна частина сучасних СУБД здатна працювати на комп'ютерах різної архітектури під управлінням різних операційних систем;

- переважна більшість сучасних СУБД забезпечують підтримку повної реляційної моделі даних, забезпечуючи цілісність категорій і цілісність на рівні посилань;

- сучасні СУБД для визначення даних і маніпуляції ними спираються на прийняті стандарти в області мов, а при обміні даними між різними СУБД базуються на існуючих технологіях з обміну інформацією;

- багато існуючі СУБД відносяться до так званим мережевим СУБД, які призначені для підтримки багато режиму роботи з базою даних і підтримки можливості децентралізованого зберігання даних;

- такі СУБД мають розвинені засоби адміністрування баз даних і засоби захисту інформації, що зберігається в них інформації;

- подібні СУБД мають засоби підключення клієнтських додатків;

- сучасні СУБД характеризуються дослідами застосування концепції фундаментальної ідеї об'єктно-орієнтованого підходу, що сприяє підвищенню рівня абстракції баз даних, що є перспективним етапом на шляху розвитку технологій баз даних [5, с.21].

База даних складається як з даних, так і з метаданих. Метадані – це дані, які описують структуру записів, що знаходяться в базі. Знаючи, як зберігаються дані, їх можна витягти. База даних є інтегрованою, так як зберігає не тільки елементи даних, але і існуючі між ними зв'язки. У базі даних метадані зберігаються в області, яка називається словником даних. У ньому описуються таблиці, стовпці, індекси, обмеження та інші компоненти, з яких складається база даних.

Для розробки програм, систем програм, які працюють з базами даних, використовуються спеціальні засоби – системи управління базами даних (СУБД).

Система управління базами даних (СУБД) – це набір програм, які використовуються для створення, адміністрування і обробки баз даних і пов'язаних з ними програм. База даних, керована такою системою, є, по суті, структурою, яку створюють, щоб зберігати в ній потрібні дані. А СУБД – це

інструмент, застосовуваний для створення такої структури і роботи з даними, які в ній зберігаються.

Сьогодні на ринку представлено безліч СУБД. Одні з них працюють тільки на великих і потужних машинах, інші на персональних комп'ютерах, ноутбуках і планшетах [11].

СУБД включає, як правило, спеціальну мову програмування і всі інші засоби, необхідні для розробки зазначених програм. В даний час найбільш відомими СУБД є: Oracle Database, MS SQL Server, MySQL (MariaDB) і ACCESS. Остання входить до складу професійного офісного пакету Microsoft Office. Це сучасні системи з великими можливостями, призначені для розробки складних програмних комплексів [12].

Яким би не був тип комп'ютера, який зберігає базу даних, а також незалежно від того, підключена чи машина до мережі, потік інформації між базою даних та користувачем, в принципі, один і той же. На рис. 1.2 зображено, як користувач взаємодіє з базою даних за допомогою СУБД. Остання приховує технічні деталі збереження даних, завдяки чому прикладній програмі приходиться мати справу тільки з логічними та характеристиками, а не зі способами збереження даних в комп'ютері.

Рис. 1.2. Система інформаційної системи, яка працює на основі СУБД

1.2. Основні поняття реляційної моделі бази даних

БД може бути заснована на одній моделі або на сукупності кількох моделей. Будь-яку модель даних можна розглядати як об'єкт, який

характеризується своїми властивостями (параметрами), і над нею, як над об'єктом, можна проводити будь-які дії [10, с.24-28].

Через те, що користувача БД не цікавить подробиці фізичного зберігання даних, в поданні даних можна виділити два рівні абстракції: логічний і фізичний. На логічному рівні описуються дані інформаційною моделлю, виділяють зовнішню і внутрішню інформаційні моделі, а на фізичному –фізичною моделлю. Інформаційна модель повинна відображати предметну область в термінах зрозумілих і звичних користувачеві. Звичайна інформаційна модель описує об'єкти будь-якої природи в термінах: сутність, атрибути та зв'язку. Абстрактна модель повинна бути описана для подання в ЕОМ. Цей опис робиться засобами моделі даних, яку підтримує СУБД і називається внутрішньої або концептуальної інформаційної схеми. Отже, СУБД підтримує деяку модель даних і відображає її до відповідних структур фізичної БД.

Модель Даних (МД) – засіб логічного представлення фізичних даних. Формалізований опис даних, що відображає їх склад, типи даних, а також взаємозв'язок між ними.

МД, яку підтримує СУБД на логічному рівні визначається:

1) допустимої структурою даних, різноманітністю та кількістю типів даних, які можна описати за допомогою МД;

2) безліччю допустимих операцій над даними;

3) обмеженнями контролю цілісності.

На рис. 1.3 подано класифікацію моделей даних:

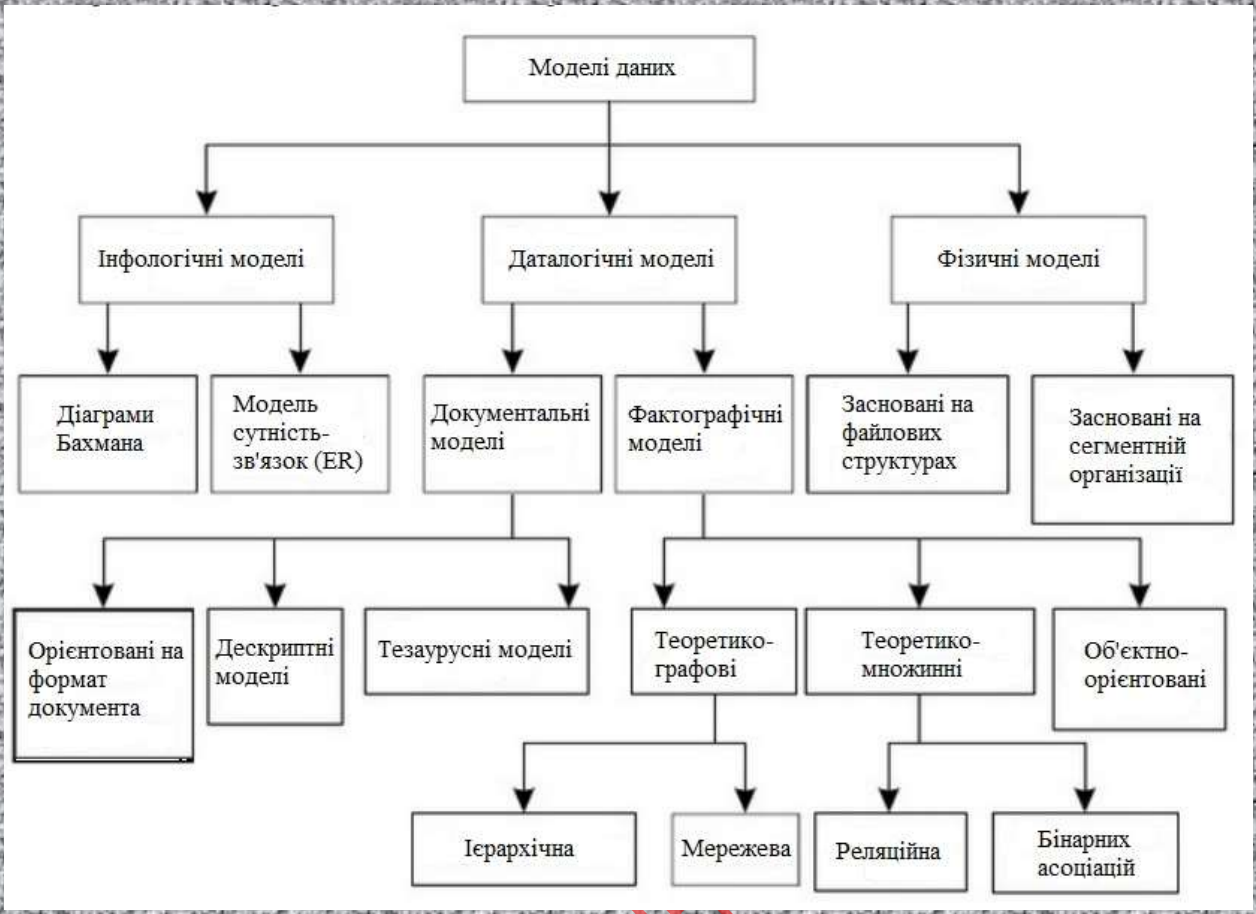


Рис. 1.3. Класифікація моделей даних

Більшість баз даних є реляційними. Творцем реляційної моделі є співробітник фірми IBM доктор Е. Ф. Кодд. Поклавши теорію відносин в основу реляційної моделі, Е. Кодд обгрунтував реляційну замкнутість відносин і ряду деяких спеціальних операцій, які застосовуються відразу до всієї множини рядків відносин, а не до окремого рядка. Зазначена реляційна замкнутість означає, що результатом виконання операцій над відносинами є також ставлення, над яким в свою чергу можна здійснити деяку операцію. З цього випливає, що в даній моделі можна оперувати реляційними виразами, а не тільки окремими операндами у вигляді простих імен таблиць [10].

Термін реляційна модель (relational model) – математична теорія, яка описує структуру даних, логіку контролю цілісності даних і правила управління даними. У найпростішому випадку вона являє собою двомірний масив або двомірну таблицю, а при створенні складних інформаційних

моделей складає сукупність взаємопов'язаних таблиць. Кожен рядок такої таблиці називається записом, а стовпець – полем [17, с.25].

У контексті опису структур даних реляційної моделі вводиться поняття нормалізованого відношення. Більш того, нормалізоване відношення є єдиною базовою структурою, якою оперує ця модель. В контексті цілісності даних мова йде про цілісність сутностей (таблиць, відносин) і зовнішніх ключів.

До переваг реляційного підходу можна віднести:

• вона є логічною, що дозволяє на досить високому рівні абстракції проектувати базу даних, а не прив'язуватись до конкретної фізичної реалізації таблиць, зв'язків та інших об'єктів (і в сумі, наприклад, вибирати конкретну СУБД вже після побудови інфологічної моделі);

• вона побудована на основі суворого математичного апарату, що дозволяє не тільки визначати застосовність, принципову можливість і коректність тих або інших операцій, але і оцінювати їх складність (і навіть служить основою для розрахунків показників продуктивності);

• вона описує як декларативний підхід, так і процедурний. Декларативний підхід (у вигляді мови SQL) реалізований майже ідентично в переважній більшості реляційних СУБД, в той час як процедурний (тригери, збережені процедури і функції) має стільки особливостей, що можна сміливо говорити про повну несумісність відповідних рішень в різних СУБД [11, с.11].

З недоліків реляційної моделі варто відзначити, що:

• вона програє іншим моделям в оптимальності опису і використання деяких структур даних;

• вона не позбавлена від типової і для інших моделей проблеми високої складності (для людини) розробки великих баз даних;

• побудовані на основі реляційної моделі бази даних при виконанні багатьох операцій є досить вимогливими до обсягів пам'яті і потужності обчислювальних ресурсів.

І все ж до сих пір у реляційної моделі немає альтернативи, яка поєднувала б у собі всі її переваги і універсальність і була б врятована від її

недоліків. Тому для по переважної більшості випадків, в яких для зберігання і обробки даних потрібна база даних, мова автоматично йде саме про реляційні бази даних [2].

Оскільки нормалізоване ставлення слугує єдиною базовою структурою, якою оперує реляційна модель даних. Однак з поняттям «відношення» нерозривно пов'язані такі терміни як «тип даних», «домен даних», «атрибут», «кортеж», «первинний ключ», які ми послідовно розглянемо в цьому розділі. «Відношення» (relation) – безліч сутностей, що володіють однаковим набором атрибутів. В контексті реляційних баз даних відношення складається з заголовка (схеми) і тіла (набору кортежів). Важливо розділяти поняття «відношення» (relation) і «зв'язок» (relationship). Строго кажучи, термін «відношення багато до багатьох» є невірним, має бути «зв'язок багато до багатьох».

У світлі щойно зробленої примітки необхідно пояснити ще одну типову термінологічну проблему. Складність в тому, що поняття «сутності» вкрай розпливчасте і часто визначається як «все, що реально існує в світі». Давайте посилимо ситуацію і формулюємо поняття відношення, отже: «відношення – це сутність, яка представляє безліч сутностей, що володіють певним набором атрибутів». Тепер варто відзначити, що і атрибут – це теж сутність. Разом: «відношення – це сутність, яка представляє собою множину сутностей, що володіють певним набором сутностей».

В контексті баз даних нас буде хвилювати тільки те, що при інфологічному (концептуальному) моделюванні, як правило, говорять про «сутності і зв'язках», а при даталогічному і фізичному – про «відношеннях і зв'язках» і «таблицях і зв'язках» [3, с.220-225].

І почнемо ми з самого фундаментального – типу даних. Тип даних (data type) – це набір об'єктів даних певної структури і набір допустимих операцій, в будь-який з яких такі об'єкти можуть виступати операндами. У загальному випадку поняття «тип даних» тут еквівалентно аналогічному поняттю в мовах програмування. Всі значення, якими оперує база даних, є типізованими (тобто

в будь-який момент часу відомий їх тип). Для більшості типів даних існують операції їх перетворення (наприклад, рядок «12.34» може бути перетворена в дробове число 12.34). На типи даних можуть застосовуватися додаткові умови, що призводить до появи поняття «домен даних». Домен даних (attribute domain) – це набір всіх можливих значень атрибута відношення.

Суть цього поняття найпростіше пояснити на прикладі. Припустимо, в деякому відношенні є атрибут «Код товару», який представлений рядком. І також відомо, що всі коди товару формуються за правилом «три букви англійського алфавіту в верхньому регістрі, чотири цифри, дві букви англійського алфавіту в нижньому регістрі» (наприклад, «ABC1234yz»). Очевидно, що рядок «понеділок» не є кодом товару, тобто не входить до відповідного домен, хоч і відноситься до того ж типу даних (до рядків) [1].

Поняття домену важливо в силу того, що дані вважаються порівнянними тільки в разі належності до одного домену, тобто безглуздо порівнювати код товару і назва дня тижня (хоча технічно таку операцію дозволить вам виконати будь-яка СУБД).

У переважній більшості СУБД доводиться використовувати тригери і для контролю приналежності даних одного домену. Не зважаючи на деяку складність такого контролю і той факт, що виконання відповідних дій знижує продуктивність деяких операцій з даними, воно дозволяє відчутно підвищити захищеність даних, знижуючи ризик передачі (і збереження) некоректних значень.

Для термінів, які були вище вказані, наведемо визначення. Атрибут (attribute) – це іменована властивість сутності (відносини). Кортеж (tuple) – це частина відношення, що являє собою унікальну взаємопов'язану комбінацію значень, кожна з яких відповідає своєму атрибуту. Ключовий атрибут (key attribute, prime attribute) – це атрибут відношення, що входить до складу як мінімум одного потенційного ключа цього відношення. Неключовий атрибут (nonkey attribute) – це атрибут відношення, що не входить до складу жодного з потенційних ключів цього відношення. Первинний атрибут (primary key

attribute) – це атрибут відношення, що входить до складу первинного ключа цього відношення. Оскільки кожний різновид ключів має свої особливі властивості, визначення ключа як такого можна сформулювати лише в загальному вигляді: ключ (key) – ідентифікатор, який є частиною набору елементів даних. В контексті реляційних СУБД ключем вважається сукупність атрибутів відносини (або окремий атрибут), що володіє певними, характерними для даного виду ключа властивостями.

На рис.1.4 представлений загальний перелік ключів і їх взаємозв'язок один з одним.

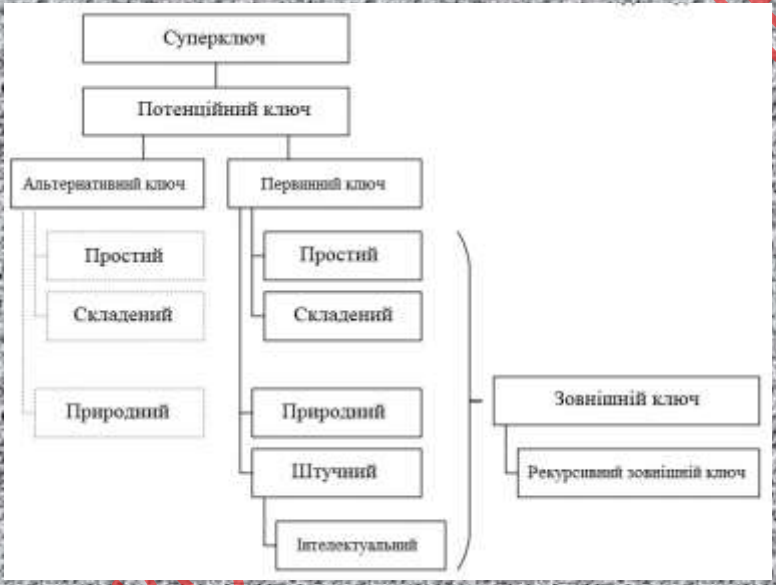


Рис.1.4. Види ключів та їх взаємозв'язок

Відразу ж варто відзначити, що штучні альтернативні ключі теоретично можуть існувати, але практично не мають ніякого сенсу (більш того - швидше шкодять, підвищуючи навантаження на СУБД).

Суперключ (superkey) – підмножина атрибутів відношення, що унікально ідентифікує будь-який кортеж. Суперключ він може містити в собі «зайві» елементи, видалення яких не приведе до втрати унікальності значень, тому що суперключ не має властивість нескоротних [1].

Потенційний ключ (candidate key) – нескоротня підмножина атрибутів відношення, що унікально ідентифікує будь-який кортеж. Іншими словами, потенційний ключ – це такий суперключ, в якому немає «зайвих» елементів.

Як правило, один з потенційних ключів в подальшому стає первинним ключем (а решта потенційних ключі стають альтернативними ключами).

Альтернативний ключ (alternate key) – потенційний ключ відношення, що не обраним в якості первинного ключа. Альтернативні ключі вимагають особливої уваги. За визначенням їх значення гарантовано ідентифікують кортежі відносини (тобто не можуть дублюватися), але СУБД (поки) нічого про це «не знає» і контролює унікальність тільки значень первинного ключа. Первинний ключ (primary key, PK) – потенційний ключ, обраний в якості основного засобу гарантованої ідентифікації кортежів відношення. Також варто відзначити, що первинний ключ повинен мати властивість мінімальності, яке слід трактувати буквально: якщо існує кілька потенційних ключів, що складаються з однакової кількості атрибутів, варто вибрати в якості первинного той потенційний ключ, який «фізично менше», тобто містить менше даних [3, с.74].

Отже, самі основні факти про первинному ключі, які важливо відзначити на даний момент:

• значення первинного ключа не можуть дублюватися (тобто вони унікальні для кожного рядка таблиці);

• первинний ключ повинен бути нескоротнім (тобто не містити «зайвих» елементів) і мінімальним (тобто в якості первинного ключа повинен бути обраний самий «короткий» потенційний ключ);

• первинний ключ використовується для гарантованої однозначної ідентифікації рядка таблиці (що випливає з властивості унікальності його значень) і для організації зв'язків;

• первинний ключ не може містити невизначене значення (NULL) – це властивість випливає з логіки забезпечення посилальної цілісності.

Традиційно на схемах відносин первинний ключ прийнято розміщувати в самому верху, тобто робити його першим полем таблиці (першими полями, якщо ключ складовою). У переважній більшості засобів проектування баз даних для «перетворення» поля в первинний ключ досить просто поставити

галочку у відповідному чек-бокс, тому що створення первинних ключів – вкрай затребувана і дуже часто виконувана операція. Забігаючи вперед варто відзначити, що найчастіше первинний ключ є кластерним індексом, тобто СУБД фізично впорядковує дані на диску за значеннями первинного ключа. Тепер розглянемо три класичних види зв'язків [1].

Зв'язок один до багатьох (one to many correspondence) – асоціація, яка об'єднує два відношення таким чином, що одному кортежу батьківського відношення (або навіть нулю таких кортежів) може відповідати будь-яку кількість кортежів дочірнього відношення.

Зв'язок багато до одного (many to one correspondence) – асоціація, яка об'єднує два відношення таким чином, будь-яку кількість кортежів дочірнього відношення може відповідати одному кортежу батьківського відношення (або навіть нулю таких кортежів).

Зв'язок багато до багатьох (many to many correspondence) – асоціація, яка об'єднує два відношення таким чином, що одному кортежу будь-якого з об'єднаних відношень може відповідати будь-яку кількість кортежів другого відношення.

Зв'язок один до одного (one to one correspondence) – асоціація, яка об'єднує два відношення таким чином, що одному кортежу батьківського відношення може відповідати не більше одного кортежу дочірнього відношення [5].

Потужність зв'язку може приймати і «нестандартне» значення, тобто крім зв'язків «один до багатьох», «один до одного» і «багато до багатьох» можуть існувати зв'язку виду «X до Y», де X і Y – довільні цілі числа або діапазони значень, наприклад, «1 до 3», « 1 до 0 ... 5 »,« 0 ... 2 до 1 ».

Через те, що індекси майже повністю лежать в області фізичного рівня проектування бази даних, тому реляційна теорія їх майже не зачіпає.

У загальному вигляді визначення поняття «індекс» може бути сформульовано таким чином: індекс (index) – спеціальна структура бази

даних, яка використовується для прискорення пошуку записів і фізичного доступу до записів.

З індексами на рівні бази даних та виконанні запитів до бази даних можна здійснити наступні основні операції:

• створити (видалити) індекс;

• відключити (включити) існуючий індекс;

• проаналізувати використання індексу при виконанні запиту;

• дати СУБД підказку по використанню індексу [12].

Індекси широко використовуються при проектуванні баз даних в силу наступних переваг:

• розмір індексів значно менше розміру індексованих даних, що дозволяє розміщувати їх в оперативній пам'яті для прискорення доступу;

• структура індексів спеціальним чином оптимізована для виконання операцій пошуку;

• як наслідок двох попередніх пунктів, індекси значно (іноді – на кілька порядків) прискорюють пошук даних.

Однак, у індексів є і недоліки, які обов'язково слід враховувати, щоб не створювати індекси там, де вони не потрібні:

• коли індексів стає багато, вони займають відчутний обсяг оперативної пам'яті;

• наявність індексів відчутно уповільнює операції модифікації даних (вставки, видалення, оновлення), тому що при зміні даних СУБД необхідно оновити індекси, приводивши їх у відповідність з новими значеннями індексованих даних.

Звідси винесемо причини корисності створення індексу:

• операції читання з таблиці виконуються набагато частіше, ніж операції модифікації;

• поле або сукупність полів часто фігурують в запитах в секції WHERE;

• дослідження показало, що наявність індексу підвищує продуктивність запитів;

• необхідно забезпечити унікальність значення поля (або сукупності полів), яка не є первинним ключем (в такому випадку: унікальний індекс);

• поле (або сукупність полів) є зовнішнім ключем – в такому випадку індекси можуть значно прискорити виконання JOIN-запитів.

Різновидів індексів дуже багато, і зараз ми розглянемо основні приклади (табл. 1.1).

Таблиця 1.1

Види індексів та їх взаємозв'язок

Індекс		
По кількості полів		**По складності ієрархії**
Простий	Складений	Однорівневий / Багаторівневий
По унікальності записів		**По відношенню з SQL-запитом**
Унікальний	Неунікальний	Покриваючий / Непокриваючий
По співвідношенню розташування записів		**По специфічним функціям**
Кластерний та некластерний	Первинний	Стовпчиковий
По структурі збереження		Зі включеними стовпчиками
Розподіленні	Нерозподілені	На обчислюваних стовпчиках
По ступеню деталізації		На значеннях функції
Щільний	Нещільний	З фільтром
По базовій структурі		Просторовий
На основі B-дерева		Повнотекстовий
На основі T-дерева		Доменний
На основі R-дерева		XML-індекс
На основі хеш-таблиці		*Постійно з'являються нові*
На основі бітової маски		
Постійно з'являються нові		

Також відзначимо, що в залежності від СУБД і обраного методу доступу список підтримуваних індексів і особливості їх реалізації можуть дуже сильно відрізнятися.

За кількістю полів індекси класифікуються абсолютно аналогічно ключам, а саме:

Простий індекс (simple index) – індекс, побудований на одному полі таблиці.

Складений індекс (composite index) – індекс, побудований на двох і більше полях таблиці [14, с.84].

Як простий, так і складений індекси можуть використовуватися для забезпечення унікальності значень альтернативних ключів (тоді це буде унікальний індекс) або прискорення пошуку записів за значенням індексованих полів (такий індекс може бути як унікальним, так і неунікальним).

Будь-яка сучасна СУБД підтримує використання як простих, так і складених індексів. Оскільки дуже часто можна почути запитання про те, як складовий індекс організований фізично, пояснимо цей момент на прикладі порівняння простого і складеного індексу на основі збалансованого дерева. На рис. 1.5 структура такого дерева представлена спрощено.

Рис. 1.5. Структура простого та складеного індексу на основі B-дерева

За унікальністю записів індекси також можна порівняти з ключами в тому сенсі, що поняття «унікальний індекс» і «ключ» навіть на рівні синтаксису SQL деякими СУБД розглядаються як синоніми. У той же час варто пам'ятати, що реляційна теорія оперує поняттям «ключ», а «індекс» з'являється на рівні реалізації бази даних в конкретній СУБД [10].

В MS SQL Server включення властивості унікальності значення поля призводить до створення на цьому полі унікального індексу (і навпаки – створення на поле унікального індексу призводить до включення властивості унікальності значень цього поля).

За співвідношенням з розташуванням записів індекси таблиці поділяються на кластерні та некластерні. Кластерний індекс (clustered index) – індекс, який побудований на полі, по якому здійснюється фізичне упорядкування даних у файлі. Первинний індекс (primary index) – індекс, який побудований на полі з унікальними значеннями, по якому здійснюється фізичне упорядкування даних у файлі. Некластерний індекс (non-clustered index) – індекс, який побудований на полі, по якому не здійснюється фізичне упорядкування даних у файлі [12].

На практиці в СУБД ситуація така: MS SQL Server дозволяє побудувати «кластерний унікальний» індекс на поле (полях), що не є первинним ключем.

Ключова відмінність полягає в тому, що первинний індекс містить дані про розташування кожного запису з унікальним значенням індексованого поля, а кластерний – про початок блоку записів з однаковими значеннями індексованого поля. Якщо не вдаватися в технічні особливості реалізації алгоритмів побудови і використання таких індексів – в іншому вони ідентичні. Некластерні же індекси відрізняються від кластерних тим, що послідовність записів в індексі і фізичному файлі не збігається. Як правило, це призводить до ускладнення структури самого індексу через необхідність зберігання кількох різних адрес записів з однаковими значеннями індексованого поля.

Тепер розглянемо основні області застосування кожного різновиду представлених в даній класифікації індексів [14].

Таблиця 1.2

Основні сфери застосування класифікації індексів

Індекс на основі:	B-дерева	T-дерева	R-дерева	Хеш-таблиці	Бітової маски
Основна сфера застосування буває:	Типова форма реалізації більшості	В основному – для in-	Просторові та	Друга по частоті використання	Стовпчики з малою потужністю

	індексів, ефективних при операціях =, >, <, які можуть завантажуватись в пам'ять частинами	memory баз даних, коли всі дані знаходяться в оперативній пам'яті	геометричні дані	(після B-дерев) форма реалізації індексів, ефективні при операціях =и!=	
Простим	Так	Так	Так	Так	Так
Складеним	Так	Так	Теоретично	Так	Так
Унікальним	Так	Так	Теоретично	Так	Так
Неунікальним	Так	Так	Так	Так	Так
Кластерним	Так	Теоретично	Теоретично	Теоретично	Так
Некластерним	Так	Так	Так	Так	Так
Роздільним	Так	Теоретично	Теоретично	Теоретично	Так
Нероздільним	Так	Так	Так	Так	Так
Щільним	Так	Так	Так	Так	Так
Нещільним	Так	Теоретично	Теоретично	Теоретично	Так
Однорівневим	Вкрай рідко	Вкрай рідко	Теоретично	Так	Так
Багаторівневим	Так	Так	Так	Теоретично	Теоретично
Покриваючим	Так	Так	Так	Так	Так
Непокриваючим	Так	Так	Так	Так	Так

Незважаючи на всю красу і міць повнотекстових індексів, варто звернути увагу на те, що вони мають широкий спектр обмежень, а їх використання вимагає точного розуміння того, що потрібно зробити. Іншими словами: чудес чекати не варто, але деякі запити можна відчутно прискорити.

1.3. Проектування баз даних на інфологічному, даталогічному та фізичному рівнях

Процес проектування бази даних являє собою послідовність переходів від словесного опису інформаційної структури предметної області до формалізованого опису об'єктів предметної області в термінах деякої моделі. У загальному випадку виділяють наступні етапи проектування:

- Інфологічний;
- Даталогічний;
- Фізичний.

Звідси ми переходимо до розгляду етапів моделювання та проектування баз даних, тобто до побудови декількох взаємопов'язаних моделей бази даних, кожна з яких покликана служити своїй особливої мети (табл.1.3).

Таблиця 1.3

Узагальнена схема рівнів моделювання бази даних

Рівень		Що описує	Чим описує
Логічний рівень	Інфологічний	Предметна область без прив'язки до виду бази даних	Сутності, атрибути, деякі зв'язки
	Даталогічний	Предметна область з прив'язкою до виду бази даних або навіть конкретної СУБД	Сутності, атрибути, зв'язки, ключі, індекси та уявлення
	Фізичний	Технічні аспекти реалізації бази даних під управлінням конкретної СУБД	Сутності, атрибути, зв'язки, ключі, індекси, уявлення тригери, збережені підпрограми, методи доступу, кодировки, права доступу

Інфологічний (концептуальний) рівень (conceptual level) моделювання ставить собі за мету створення концептуальної моделі (conceptual model), що відбиває основні сутності предметної області, їх атрибути та зв'язку між сутностями. Для її досягнення виконується ряд поставлених процедур:

1. Визначення сутностей та їх документування;
2. Визначення зав'язків між сутностями та їх документуванням;
3. Створення ER-моделі предметної області;
4. Визначення атрибутів та їх документування:
   - Ім'я атрибута та його опис;
   - Тип та розмірність значень;
   - Значення, що приймається для атрибута по замовчуванню;
   - Чи може атрибут мати NULL-значення;

- Чи являється атрибут складеним, а якщо так, то з яких атрибутів він складається.

Як предметну область цілком, так і безпосередньо дані на цьому рівні можна описувати різними моделями (наприклад, семантичною, графовою, «сутність-зв'язок») [21]. Як спосіб представлення моделі найчастіше використовується UML або простий словесний опис (що особливо зручно для обговорення моделі з представником замовника, які не є IT-фахівцем). Дослідження предметної області як раз і представляє основну складність, яка поділяється на такі локальні проблеми:

• вилучення інформації (універсальна проблема для будь-якого проекту (навіть поза контекстом баз даних). Як правило, у замовника немає готового повного технічного опису проекту, тому необхідно організувати збір вимог до проекту, і в них виділити ті вимоги, які прямо або побічно належать до бази даних);

• глибина дослідження (найбільша складність для початківців проектувальників, що особливо добре помітно на прикладі навчальних робіт. Недостатньо виявити лише деякі сутності і деякі їх атрибути. Необхідно виявити їх всі);

• межі дослідження (проблема часто не технічна, а управлінська, але від цього вона не стає простішою: в силу наполягання замовника або з технічних міркувань в базу даних можуть почати потрапляти сутності, цілком реально існуючі і пов'язані з предметною областю, але не актуальні для реалізованого проекту).

До переліку завдань даного рівня проектування також можна сміливо додати забезпечення відповідності бази даних ключовим вимогам: і нехай їх неможливо забезпечити одним лише проектуванням на інфологічному рівні, саме тут закладається фундамент для багатьох рішень, які будуть прийняті на наступних рівнях і дозволять створити дійсно ефективну базу даних.

У плані такої зручності неперевершеними залишаються всім зрозумілі і звичні багаторівневі списки, які можна відобразити в будь-якому текстовому редакторі. Першим рівнем будуть сутності, другим – їх атрибути.

Текстове представлення є зручним не тільки своєю звичністю, а й своєю практичністю:

• у всіх на комп'ютерах є необхідне програмне забезпечення для перегляду і редагування текстових документів;

• створення і правка таких документів відбувається дуже швидко;

• можна дуже зручно розташовувати позначки і коментарі (які будуть видні відразу без додаткових дій);

• результат може бути миттєво роздрукований.

У текстового представлення є ряд серйозних недоліків:

• воно незручно для подання взаємозв'язку сутностей;

• воно не компактне (може займати кілька десятків сторінок там, де інші форми подання зайняли б пару екранів);

• воно допускає різночитання технічних аспектів реалізації бази даних (а найчастіше ці аспекти і зовсім виявляються втрачені);

• спроби усунути зазначені вище недоліки роблять текстове представлення перевантаженим інформацією і поступово зводять нанівець його переваги [22, с.35].

Альтернативами текстовому поданням є графічні форми – у вигляді семантичних моделей, графових моделей і UML-діаграм. Семантичні і графові моделі поширені не так сильно, як UML-діаграми, але виявляються дуже корисними при описі складних взаємозв'язків в предметних областях. Їх складно використовувати «безпосередньо» для проектування баз даних (вони не проектуються безпосередньо на реляційну модель), але в якості постійної доступної «шпаргалки» вони майже незамінні.

Даталогічний рівень (часто його називають просто «логічним», logical level) моделювання деталізує інфологічну модель, перетворюючи її в логічну схему (logical schema), на якій раніше виявлені суті, атрибути та зв'язку

оформляються згідно з правилами моделювання для обраного виду бази даних (можливо, навіть з урахуванням конкретної СУБД).

Для її досягнення виконуються наступні процедури:

1. Вибір моделі даних;
2. Визначення набора таблиць, виходячи з ER-моделі та їх документування;
3. Нормалізація таблиць;
4. Перевірка логічної моделі даних на предмет можливості виконання усіх транзакцій, передбачених користувачем. Транзакція – це набір дій, що виконуються окремим користувачем або прикладною програмою з метою зміни вмісту бази даних.
5. Визначення вимог підтримки цілісності даних та їх документування;
6. Створення остаточного варіанта логічної моделі та обговорення його з користувачами.

У багатьох засобах проектування баз даних, що підтримують поділ лише на «логічне» і «фізичне» проектування саме цей рівень називається «логічним». Тут вже з'являються цілком звичні таблиці, поля, ключі, зв'язки, частина індексів і уявлень – все те, що і складає суть бази даних. Оскільки реалізація цих об'єктів залежить від виду бази даних (і навіть конкретної СУБД), тут вже не можна будувати строго абстрактну модель, і доводиться враховувати типові особливості обраних бази даних і СУБД. Як спосіб представлення моделі на цьому рівні найчастіше використовуватиметься UML або поступово втрачають популярність нотація IDEF0 [19].

Для інфологічного рівня нам в загальному випадку було достатньо будь-якого текстового редактора (або будь-якого зручного нам і представникам замовника спеціалізованого редактора UML (або іншої графічної мови) – таких інструментів багато, і більшість фахівців добре вміє ними користуватися).

Для даталогічного рівня нам потрібні спеціалізовані інструменти, що дозволяють оптимальним чином формувати структуру бази даних,

багаторазово її переглядати і аналізувати, а в кінцевому підсумку і експортувати в повноцінний SQL-код для імпорту в СУБД [32]. Кожен виробник СУБД традиційно надає свої власні інструменти, наприклад:

• MySQL Workbench198 для MySQL.

• SQL Server Management Studio199 для MS SQL Server.

• SQL Developer Data Modeler200 для Oracle.

Це, безумовно, дуже потужні інструменти, створені з урахуванням всіх особливостей цільової СУБД, і вони однозначно заслуговують на увагу. Якби не одне «але»: вони більше знадобляться вам для проектування на наступному, фізичному рівні.

Фізичний рівень (physical level) моделювання продовжує деталізацію і дозволяє створити фізичну схему (physical schema), на якій максимально враховуються технічні особливості роботи конкретної СУБД і її можливості з організації та управління структурами, розроблюваної бази даних і даними в ній.

Процедури фізичного проектування наступні:

1. Проектування таблиць бази даних засобами обраної СУБД;
2. Реалізація бізнес-правил в сфері обраної СУБД;
3. Проектування фізичної організації бази даних;
4. Розробка стратегії захисту бази даних;
5. Організація моніторингу функціонування бази даних та її налаштування.

На цьому рівні модель даних може бути представлена так само, як і на попередньому – найчастіше, у вигляді UML, але однією лише графічної форми тут недостатньо, тому в хід йдуть SQL-скрипти, словесні описи необхідних змін і налаштувань, фрагменти конфігураційних файлів, підготовлені cmd/bash-скрипти, reg-файли.

Можна виділити такі загальні цілі і завдання, тому що особливий інтерес для нас представляють:

• права доступу (у відносно невеликих і простих проектах передбачається, що взаємодія з СУБД йде від імені одного користувача – працює з СУБД додаток завжди авторизується з використанням однієї і тієї ж пари «логін-пароль», і саме контролює права своїх користувачів);

• кодування (однією з найбільш частих помилок початківців при проектуванні баз даних є недостатня увага кодувань символів. Якщо при проектуванні СУБД не вказати явним чином кодування бази даних, то при перенесенні моделі бази даних в СУБД всі кодування будуть виставлені «за замовчуванням», тобто взяті з налаштувань тієї СУБД, в яку імпортується база даних);

• методи доступу (якщо покладатися на налаштування за замовчуванням, легко опинитися в ситуації, коли поведінка СУБД буде зовсім не такою, яка очікується. Можуть виникнути проблеми з продуктивністю, контролем посилальної цілісності (так, до сих пір існують методи доступу, які не підтримують такий контроль), транзакціями, блокуванням таблиць при операціях модифікації, реплікаціями, масштабуванням. Метод доступу цілком визначає поведінку бази даних на найнижчому рівні – на рівні файлів, в яких зберігаються дані);

• індекси (існує дуже широкий вибір варіантів як різновидів індексів, так і їх параметрів. Багато індексів (наприклад, унікальні або на полях, через які очевидно дуже часто буде виконуватися пошук), як правило, видно ще на даталогічному рівні моделювання);

• налаштування СУБД (в більшості випадків налаштування СУБД за замовчуванням «оптимізовані» під широкий спектр типових рішень, до яких, швидше за все, і відноситься база даних).

Як мінімум, управління кодуваннями, методами доступу і індексами є практично в будь-якому інструменті моделювання на даталогічному рівні [15]. Причому ті інструменти, які на даталогічному рівні були відзначені як надмірно вузькоспеціалізовані, тут, навпаки, можуть виявитися найбільш корисними. Що стосується управління правами доступу і налаштуваннями

СУБД, то найвигіднішим інструментом тут буде такий, який дозволить автоматично налаштовувати ці параметри кожен раз при генерації бази даних або розгортанні її в СУБД. Тобто такий інструмент повинен вміти автоматично виконувати в потрібний момент часу набір SQL-запитів, модифікувати текстові файли, вносити зміни до реєстру Windows.

Висновки до розділу 1

В першому розділі магістерської роботи було досліджено теоретичні аспекти основ баз даних та системи управління базами даних, що дало змогу отримати наступні суттєві результати та зробити наступні висновки:

1. Охарактеризовано сутність основ баз даних та систем управління базами даних. Досліджено, що бази даних та їх системи управління використовуються для роботи з великим об’ємом даних та великої кількості користувачів, де можна зберігати, обробляти і змінювати інформацію у великих обсягах. Таким чином визначено що, база даних – це велика сукупність даних, підлеглих додатковим правилам. Такі правила залежать від виду бази даних, а за їх дотримання відповідає система управління базами даних. СУБД – це спеціальне програмне забезпечення, призначене для управління базами даних, де взаємодія з такими СУБД відбувається шляхом виконання SQL-запитів.

2. Досліджено основні поняття реляційної моделі бази даних, яка проектується структурованою мовою запитів (Structured Query Language, SQL) для визначення і обробки даних. Проаналізовано, що SQL одна з найбільш гнучких і поширених мов запитів, через те, що її вибір дозволяє мінімізувати ряд ризиків. Охарактеризовано, що реляційні БД зберігають інформацію в таблицях, і основна їх робота крутиться навколо відношень цих таблиць один до одного. У таблицях зберігається інформація про об'єкти, представлених в базі даних. У кожному стовпчику таблиці зберігається певний тип даних, в

кожному осередку – значення атрибута. В ході проведення дослідження було визначено що, кожен рядок в таблиці має бути позначений унікальним ідентифікатором, так званим первинним ключем, а рядки з декількох таблиць мають бути пов'язані за допомогою зовнішніх ключів. До цих даних можна отримати доступ багатьма способами, і при цьому реорганізовувати таблиці БД не потрібно.

3. Розглянуто основні етапи проектування реляційної бази даних. Визначено, що база даних розробляється трьома основними етапи або так званими рівнями розробки архітектури: інфологічний, даталогічний та фізичний рівні моделей баз даних. Зокрема з’ясовано, що метою інфологічного проектування є створення структурованої інформаційної моделі, для якої розроблятиметься БД. Під час проектування на інфологічному рівні створюється інформаційно логічна модель, яка має відповідати таким вимогам: коректності схеми БД; простоті і зручності використання на наступних етапах проектування; описання мовою, зрозумілою проектувальникам БД, програмістам, адміністратору і майбутнім користувачам. Мета проектування на даталогічному рівні – створити структуру майбутньої бази даних, а завданням – зробити створювану структуру максимально якісною, позбавленої знайдених на даному етапі проблем. Фізичний рівень залежить як від програмного забезпечення, так і від обладнання. Вважається, що на цьому рівні визначається різні типи записів даних, існуючі індекси, способи подання полів та фізичну впорядкованість записів.

РОЗДІЛ 2

ПОБУДОВА ТА ПРОЕКТУВАННЯ РЕЛЯЦІЙНОЇ МОДЕЛІ ПРЕДМЕТНОЇ ОБЛАСТІ

2.1. Аналіз предметної області бібліотеки «Світогляд»

У різних сферах людської діяльності широкого поширення набули технології, що використовують бази даних для систематизації та зберігання виробничої інформації. Це обумовлено тим, що цей розділ інформаційних технологій має значний степінь впровадження і на практиці досить просто інтегрується під кожен конкретний випадок. До того ж в даний час існує чимало всіляких варіантів реалізації баз даних (БД) і систем управління базами даних (СУБД).

База даних – це інформаційна модель, метою створення якої є впорядковане зберігання інформації, яка має однаковий набір властивостей [21].

Система управління базами даних – це популярний інструментальний засіб, що застосовується для створення і роботи з БД великих обсягів. СУБД, що працюють з реляційними типами баз даних, в більшій мірі використовуються на персональних комп'ютерах. СУБД дозволяють організувати, структурувати і систематизувати інформацію таким чином, щоб процес зберігання і обробки на комп'ютері був в достатній мірі простий і зручний [22, с.20].

Процес проектування бази даних являє собою послідовність переходів від словесного опису інформаційної структури предметної області до формалізованого опису об'єктів предметної області в термінах деякої моделі.

Історико-культурна та наукова спадщини в нашому житті відіграють невід’ємно важливу роль. Для того щоб створити повноцінне, духовне

суспільство, потрібно не тільки мати велике бажання, а й володіти знаннями в історії, науці, літературі та інших життєвих сферах. Саме розуміння минулого, цінності культури і наукових ідей дозволяють нам залишатися людьми і утримують від тваринних інстинктів. Значну частину цих знань ми отримуємо ще на шкільній лаві, але більш глибокий розвиток і духовну їжу дарують нам музеї і бібліотеки [35].

Бібліотека – це культурно-просвітнє та науково-допоміжна установа, що організовує громадське користування творами друку. Бібліотеки систематично займаються збором, зберіганням, пропагандою та видачою читачам творів друку, а також інформаційно-бібліографічної роботою. Це інформаційний, культурний, освітній заклад, що має організованим фондом документи і представляє їх у тимчасове користування абонентам, а також здійснює інші бібліотечні послуги. Основне завдання бібліотек – це збір і зберігання друкованих видань для громадського використання.

Умовно їх можна поділити на публічні, тобто призначені для читачів різного віку і професій, і наукові, які створюються з метою зберігання інформації в різних напрямках і секторах знань.

Діяльність бібліотеки складається з трьох основних складових: зберігання, пошук, і видача на певний час, наявних в наявність уже книг. Бібліотеки систематично займаються збором, зберіганням, пропагандою та видачою читачам творів друку, а також інформаційно-бібліографічної роботою, що є загальнодоступним джерелом знань і основною базою для самоосвіти. Основними напрямками роботи будь-якої бібліотеки є: комплектування, організація книжкового фонду та обслуговування читачів.

Комплектування фондів бібліотеки складається з систематичного виявлення (шляхом перегляду бібліографічних джерел і літератури) потрібних для даної бібліотеки видань і придбання їх. Від своєчасності і повноти комплектування бібліотеки в значній мірі залежить рівень обслуговування читачів.

Організація книжкового фонду включає питання обліку, розстановки, зберігання літератури і доставки її читачеві. Правильна організація фонду полегшує читачеві користування літературою, бібліотекарю – швидке виконання читацьких вимог, а також забезпечує збереження фондів як суспільної власності [28].

Обслуговування читачів бібліотеки здійснюється різними шляхами, а саме:

- видача літератури, як в читальний зал, так і за межі бібліотеки;
- допомога окремим читачам і установам в підборі необхідної їм літератури;
- розкриття книжкових фондів бібліотеки через систему бібліотечних каталогів;
- складання інформаційно-бібліографічних посібників різного типу;
- пропаганда найбільш цінної літератури;
- репродукування текстів на замовлення читачів і т.д.

Бібліотечна діяльність пов'язана з урахуванням великої кількості операцій, безліч книг і читачів серйозно уповільнюють роботу бібліотекарів. Складність пошуку потрібної книги в каталозі займає тривалий час, і цілком спирається на компетентність працівників бібліотеки.

Більшість людей вважає, що в бібліотечну справу йдуть ті, хто любить читати. Але це все ще залишився стереотип з минулих літ. Робота в бібліотечній сфері дуже різноманітна і пов'язана, в першу чергу, з інформацією. Але як раз таки завдяки бібліотекам ми можемо знайти різні документи і навчальні посібники в електронному варіанті, саме бібліотеки займаються їх розміщенням в мережі.

Тим не менш, бібліотеки по всьому світу зараз перебувають на переломному етапі, вони концептуально змінюються, реорганізовуються, вишукують можливості зробити свої послуги корисними для сучасного користувача, реагують на виникаючі інформаційні потреби. Зі сховища

документів бібліотека перетворюється на місце, де формуються нові знання. Що стосується університетських бібліотек, вони завжди були найбільш динамічними. Тому що це робота, перш за все, з молоддю. Бібліотеки підлаштовуються під інтереси студентів, намагаються швидко реагувати на зміни в сфері освіти.

Бібліографічна діяльність здійснюється в бібліотеках всіх типів, незалежно від складу читачів і профілю роботи. Основне завдання співробітників бібліотеки – зробити звернення читачів зручним і комфортним. Це означає своєчасний і якісний підбір потрібної літератури для читача. Бібліотеки відіграють величезну роль не тільки для студентів ВНЗ в навчальному процесі, але й також для звичайних читачів, які постійно бажають вдосконалювати свої набуті знання, а також вивчати для себе абсолютно нову інформацію з метою розвитку своїх думок, надання безкінечних знань та уроків, зберігаючи при цьому наш розум активним [17].

Обсяг інформації в бібліотечних фондах безперервно росте. У зв'язку з цим, існує проблема підвищення якості та ефективності роботи бібліотек. Для цього створюються електронні бази даних, з допомогою яких можна швидко здійснювати пошук потрібної книги для студента, викладача та звичайного читача. Так само в базі даних можна відображати всіх користувачів, терміни повернення книг, кількість примірників книг і багато іншого. Якщо база даних розроблена вдало, то можна легко керувати роботою бібліотеки. Також можна складати різноманітні запити, які дозволяють відображати потрібну інформацію. Нині існує велика кількість електронних каталогів різних бібліотек. У даній роботі розглядається процес створення структури бази даних для типової бібліотеки.

База даних повинна забезпечувати автоматизацію наступних процесів: пошук, додавання даних, видалення і зміна наявної інформації про книги, читачів і співробітників. Крім того, процес пошуку повинен бути максимально спрощений: організований за ключовим словом або букві.

Справжня БД повинна являти собою інформаційну систему забезпечення діяльності трьох груп користувачів: директора, співробітників і користувачів (в даному випадку читачів) бібліотеки.

У базі даних повинно забезпечуватися зберігання і регулярне оновлення такої інформації:

- інформація про книги, зокрема: найменування, автор, рік видання і жанр;
- найменування жанрів книг;
- ПІБ авторів книг;
- інформація про читачів, таку як: ПІБ читача, дата народження читача, адреса, контактний номер телефону, а також паспортні дані;
- інформація про видані книгах: дата видачі, дата повернення, позначка про повернення, читач, на руках у якого книга і співробітник, який її видав.

Склад функцій, що забезпечують діяльність типового користувача – співробітник.

Співробітник бібліотеки повинен мати можливість виконувати за допомогою форм введення і пошукових форм наступні дії: пошук читачів, реєстрацію нових або оновлення даних про вже існуючих користувачів бібліотеки, а також пошук книг і оформлення замовлення на поповнення бібліотеки.

База даних забезпечує виконання наступних типових запитів співробітника: виведення списку доступних книг і тих, які в даний момент знаходяться у читача; відображення картотеки читачів; пошук читача по імені; висновки списку виданих книг з уточненням безпосередньо про людину, у якої вона знаходиться; угруповання книг за жанрами; розрахунок часу, яке книга провела на руках; пошук книг і автоматичне оформлення замовлення на поповнення бібліотеки.

Склад функцій, що забезпечують діяльність типового користувача – директор.

Крім функцій співробітників, директор бібліотеки повинен мати можливість додавати, редагувати і видаляти за допомогою форм введення інформацію про співробітників і посадах, в тому числі змінювати суму окладу.

Директор повинен мати можливість виконувати наступні запити: пошук співробітників, перегляд штату співробітників, висновок інформації про заробітну плату співробітників [35, с.12].

Склад функцій, що забезпечують діяльність типового користувача – читач.

Читач повинен мати можливість виконувати за допомогою пошукових форм наступні дії: пошук книг. Для користувача повинні бути заборонені функції додавання, видалення і зміни інформації.

У магістерській роботі здійснюється проектування і розробка програми на основі баз даних для предметної області, а саме бібліотеки «Світогляд». В деякій бібліотеці зберігаються книги. Кожна книга має назву, відноситься до одного жанру, має одного або декількох авторів, видана в певному видавництві. У бібліотеці може бути кілька примірників однієї книги. Кожен екземпляр має унікальний шифр. Користувачами бібліотеки є читачі, про які в картках зберігається інформація про прізвища, адресу і телефон. кожен читач може взяти кілька примірників книг, які є в наявності. При цьому в особовій картці читача зберігаються відомості про взятої книзі і датою її отримання. При поверненні книги в картці читача вказується дата повернення, і даний екземпляр книги стає доступним для інших користувачів.

Побудова бази даних бібліотеки «Світогляд» дозволяє виконувати такі дії: зберігати інформацію про книги, читачів, співробітників і виданих книгах. БД повинна забезпечувати гнучкий пошук книг. Примірники книг зберігаються у відділах, де для кожного екземпляра можна дізнатися, в якому відділі він знаходиться (або кому видано на руки). На кожен екземпляр ставиться інвентарний номер. В БД зберігається історія переміщення примірників між відділами.

Читачі реєструються в бібліотеці на певний термін (також вони можуть виписатися за власним бажанням). Коли читач бере або повертає книги, цей процес оформлюється бібліотекаром, інформація про це заноситься в БД та з часом відправляється в архів. Іноді читачі втрачають книги, інформація про втрати повинна також зберігатися в БД. Також екземпляри книг можуть бути знищені, продані або передані іншим бібліотекам (в БД необхідно тільки відзначати факт вилучення примірника книги з фонду із зазначенням причини).

2.2. Створення БД бібліотеки «Світогляд» по технології allfusion process та data modeler

В даний час для створення інформаційних систем (ІС) застосовують як об'єктно-орієнтовану методологію на основі стандарту UML, так і структурну на основі стандарту IDEF [1, 2, 3]. Мова UML, яка з'явилась набагато пізніше IDEF, ще не витіснилась остаточно із застосування IDEF. CASE-засоби компанії CA Allfusion Process та Data Modeler, які підтримують стандарт IDEF, є популярними в середовищі розробників та продовжують розвиватися і вдосконалюватись. Можливості експорту бізнес-моделей з BPwin в сферу моделювання даних ERwin відсутні як у Rational Rose, так і у Power Designer [2, 3, 4]. Розглянемо для прикладу завдання розробки ІС для бібліотеки «Світогляд».

Одним з напрямків поліпшення роботи невеликих, але численних бібліотек є впровадження досягнень інформаційних технологій [5]. Кожна бібліотека, як правило, має в наявності тисячі книг і обслуговує сотні читачів з різноманітними запитами. Для управління діловодством в таких бібліотеках потрібні прості у використанні і корисні в застосуванні ІС. Покажемо, як можна створити базу даних (БД) ІС бібліотеки із застосуванням Allfusion Process та Data Modeler.

Розробка БД включає два етапи: 1. Функціональне моделювання в BPwin діяльності бібліотеки в стандарті IDEF0 і експорт стрілок в сферу ERwin [1, 4]. 2. Створення логічної і фізичної моделей даних в ERwin на основі IDEF1x, кодогенерацію і перетворення моделей в БД на сервері MS SQL Server.

Розглянемо завдання моделювання бібліотеки «Світогляд» у вигляді такої контекстної діаграми (див. Рис. 2.1). Дана бібліотека обслуговує запити читачів з наявних книжкових фондів і проводить планові заходи під контролем міської ради. Необхідні деталі цієї діяльності розкриваються на дочірніх діаграмах за допомогою декомпозиції робіт. Варто звернути увагу, що імена всіх стрілок робіт (особливо, вхідних і вихідних) визначають претендентів на включення до складу сутностей [4].

Рис. 2.1. Контекстна діаграма IDEF0 бібліотеки «Світогляд»

Після проведення декомпозиції контекстної діаграми можна побачити основні роботи бібліотеки: «Реалізація щорічної програми діяльності», «Реєстрація читача», «Обслуговування читацького запиту», «Реєстрація нових книг» і «Реєстрація нових періодичних видань» (див. Рис. 2.2). Також на

даному рівні можна простежити обов'язки співробітників бібліотеки. Бібліотекар реєструє нових клієнтів і займається прийомом та видачою книг і періодики. Завідувач сектором комплектування та обробки літератури також бере участь в обслуговуванні читачів, надаючи їм бажану інформацію, бібліографічні списки і т.д. Крім того, він реєструє нові надходження літератури, виконує бібліографічну роботу. Директор бібліотеки контролює виконання щорічної програми, а бібліотекари і завідувач сектором комплектування, виконуючи свої прямі обов'язки, тим самим беруть участь у її реалізації.

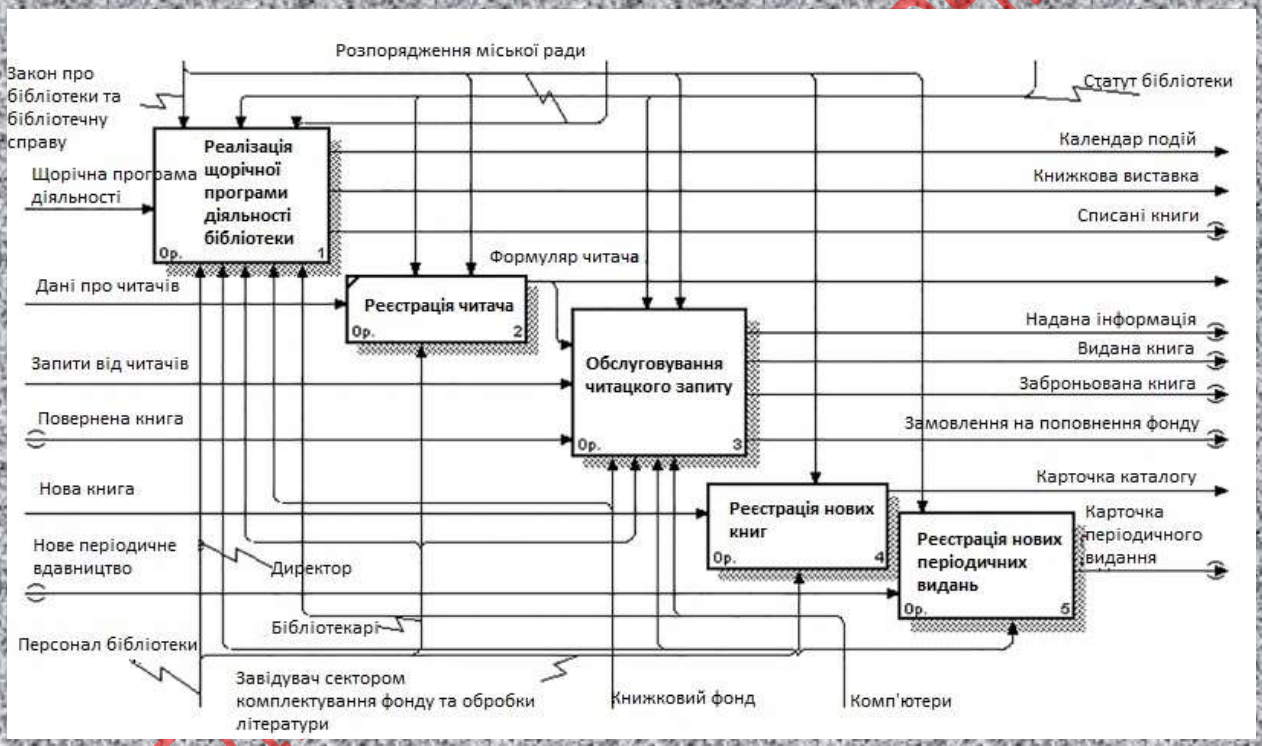


Рис. 2.2. Модель діяльності бібліотеки на верхньому рівні декомпозиції

Детальне уявлення роботи «Обслуговування читацького запиту» відбувається на наступному рівні (див. мис. 2.3).

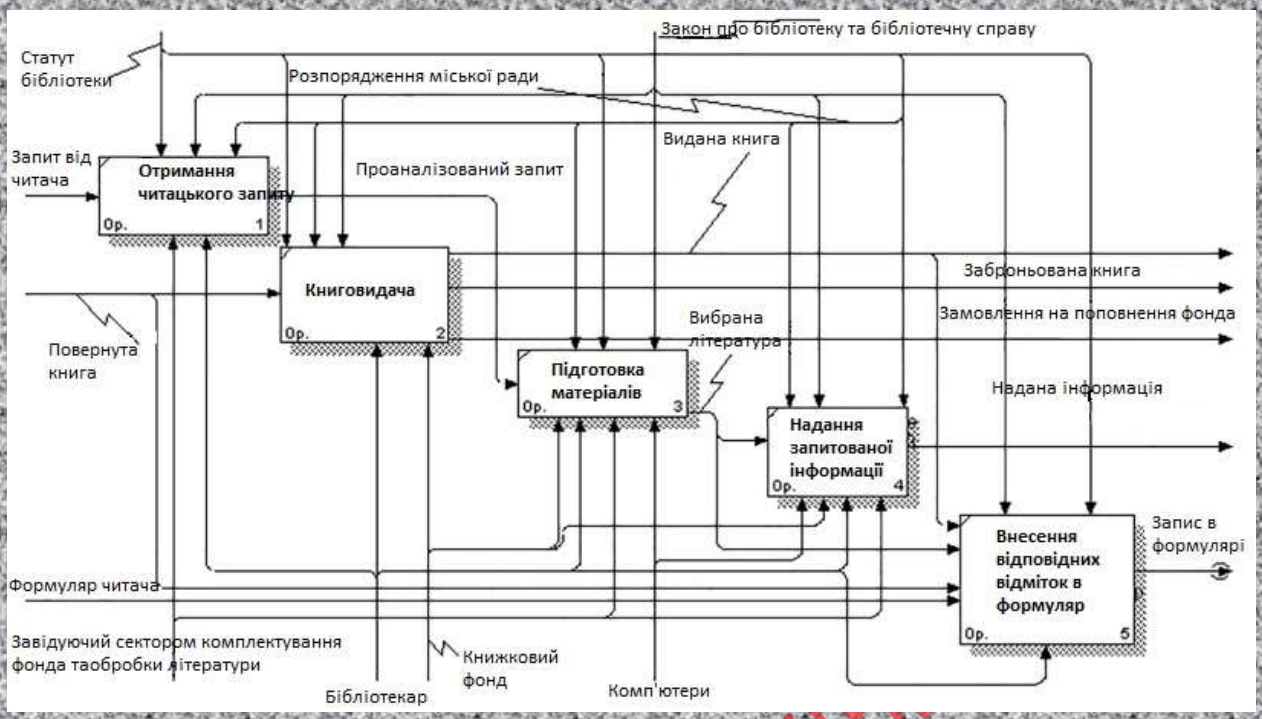


Рис. 2.3. Декомпозиція роботи «Обслуговування читацького запиту»

Отриманий від читача запит (усний чи письмовий) спочатку аналізується, а потім співробітники приступають до його виконання. Видачу літератури або її прийом проводить бібліотекар, фіксуючи факт отримання або повернення книги та періодичного видання в читацькому формулярі. У разі, якщо викликаний екземпляр фонду знаходиться в іншого читача, то бібліотекар оформляє бронювання або вносить зазначену книгу в замовлення на поповнення фонду. Повне уявлення про глибину і повноту декомпозиції дає діаграма дерева вузлів на рис. 2.4.

Рис. 2.4. Діаграма дерева вузлів для моделі IDEF0 бібліотеки «Світогляд»

У тому випадку, коли замовник вважає досить повним комплект отриманих діаграм IDEF0, можна виконувати фільтрацію і експорт імен стрілок в ERwin. Після проведення експорту виходить набір незв'язаних сутностей без атрибутів – модель складу даних [4]. Склад і число атрибутів кожної сутності визначається практичними інтересами замовника.

2.3. Розробка, проектування і реалізація інфологічної, даталогічної та фізичної моделей БД

База даних для бібліотеки «Світогляд» створюється з метою економії часу при пошуку книг, а також швидкого отримання відповідей на такі питання, як:

- наявність або відсутність даної книги в особистій бібліотеці;
- де знаходиться пошукова книга;
- хто автор пошукової книги;
- які книги даного автора зібрані в бібліотеці;

- скільки примірників цієї книги є в бібліотеці;

- в якому році і де видана шукана книга.

При роботі з базою отримувати необхідну інформацію набагато простіше. БД бібліотеки повинна забезпечувати гнучкий пошук книг (по ПІБ автора, за роком випуску, за назвою видавництва). При розробці структури бази даних бібліотеки «Світогляд» була використана концептуальна модель побудови бази даних.

Проведений аналіз предметної області та моделювання бізнес-процесів дозволяє побудувати концептуальну модель. Необхідно виділити сутності з їх атрибутами і визначити смислові зв'язки між ними [31]. Сутність – це щось таке, про що потрібно зберігати інформацію в базі даних. Сутності можуть моделювати конкретні або абстрактні поняття. Записи про певні параметри кожної з сутностей називаються атрибутами.

Концептуальна модель будується або у вигляді діаграми «Сутність-Зв’язок» (Entity-Relationship-діаграми, ER-діаграми), або записується на мові концептуального (інфологічного) моделювання (МКС, ЯІМ).

До сутностей предметної області відносяться: Читачі, Співробітники, Книги, які відображені в таблиці 2.1.

Таблиця 2.1

Сутності предметної області, а саме бібліотеки «Світогляд»

Атрибут / Сутності				
№	**Читачі**	**Співробітники**	**Книги**	**Посада**
1.	Код читача	Код співробітника	ISBN (код) книги	Код посади
2.	Прізвище читача	Прізвище співробітника	Назва книги	Назва посади
3.	Ім'я читача	Ім'я співробітника	Автор книги	Оклад
4.	По батькові читача	По батькові співробітника	Видавництво книги	

5.	Дата народження читача	Дата народження співробітника	Рік видання книги	
6.	Звідки читач	Звідки співробітник	Жанр книги	
7.	Телефон читача	Телефон співробітника		
8.	Паспортні дані читача	Паспортні дані співробітника		
9.		Посада співробітника		

Тепер можемо відобразити всі сутності та їх зв’язки у вигляді діаграми «Сутність-Зв'язок» (ER-діаграми).

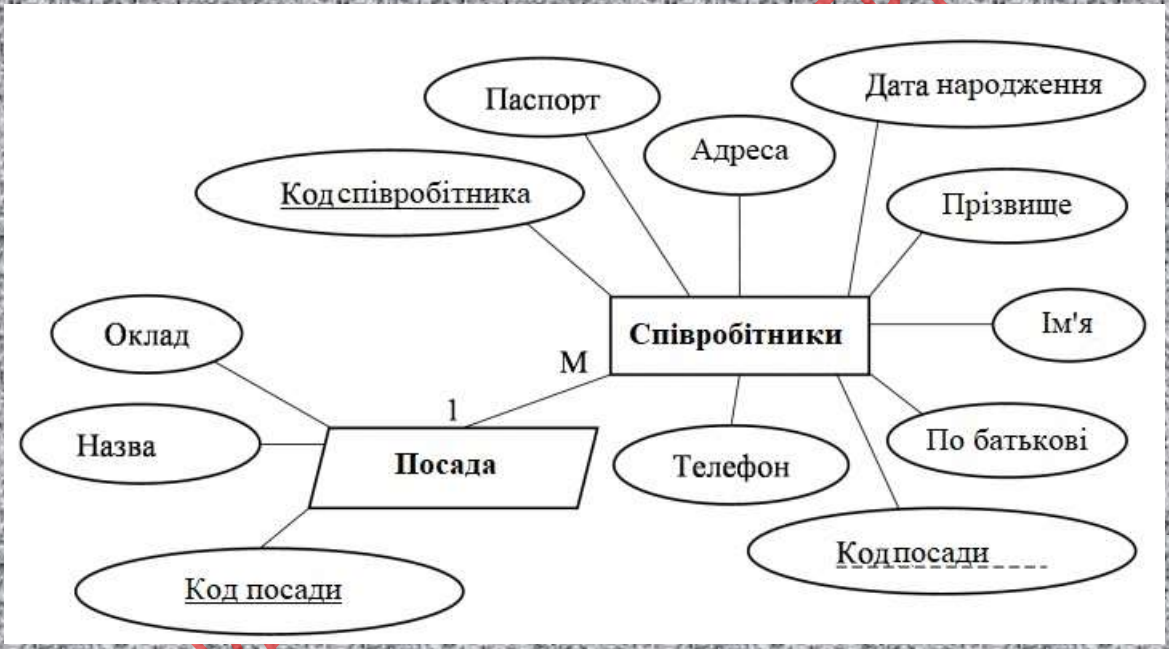


Рис. 2.5.ER-діаграма сутності «Співробітники»

Виділимо із сутностей «Книги» сутності-позначення: Автор, Видавництво та Жанр, що зображені в таблиці 2.2.

Таблиця 2.2

Сутності предметної області, а саме бібліотеки «Світогляд»

Атрибут / №	Сутності		
	Автор	**Видавництво**	**Жанр**

1.	Код автора	Код видавництва	Код жанру
2.	Прізвище автора	Назва видавництва	Назва жанру
3.	Ім'я автора	Звідки видавництва	

На основі даних з таблиці 2.2 відобразимо це в вигляді діаграми «Сутність-Зв'язок» (ER-діаграми).

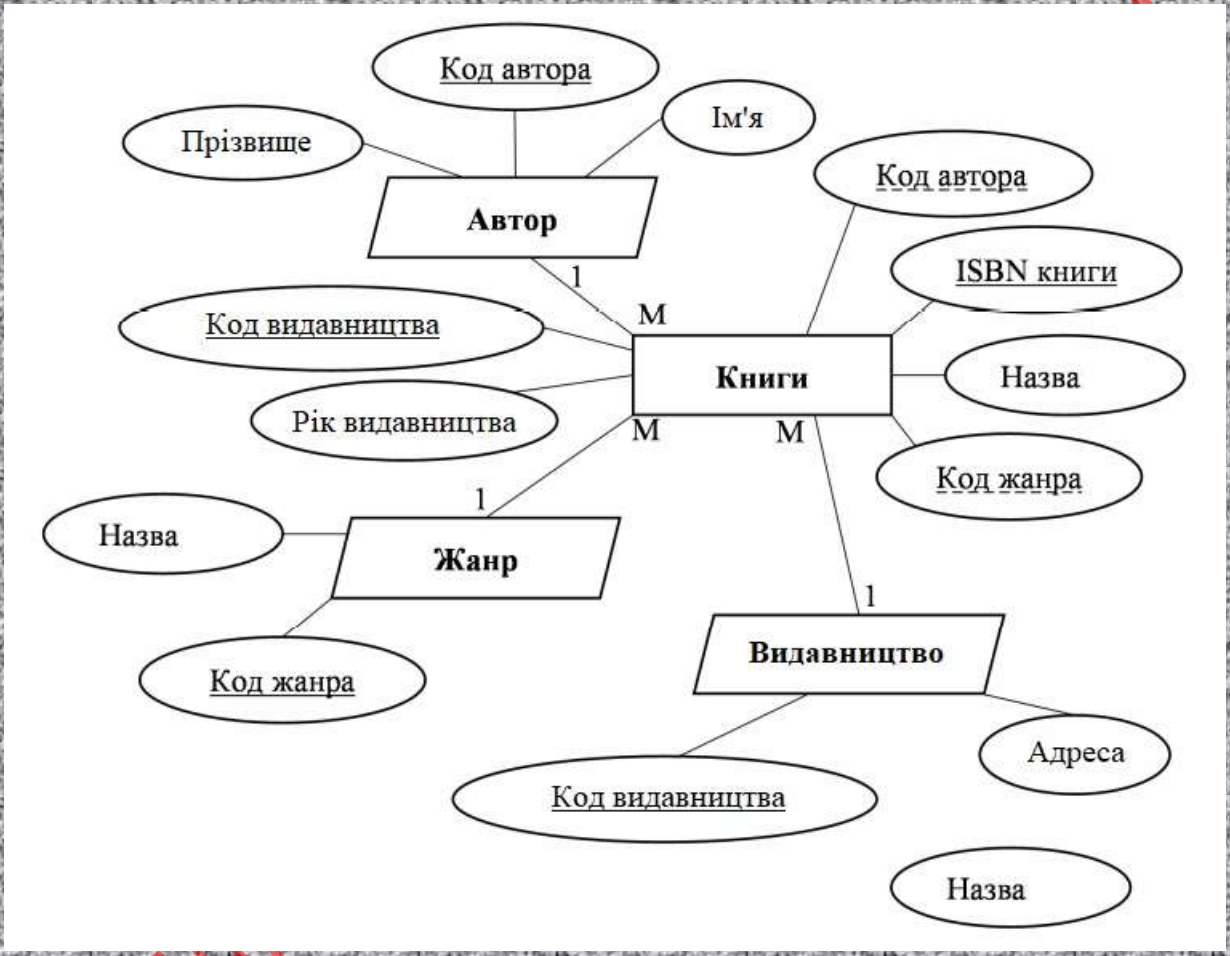


Рис. 2.6. ER-діаграма сутності «Книги»

Між сутністю «Читачі» і сутністю «Книги» є смисловий зв'язок: «Читачі беруть книги», які є відображені на рис.2.7.

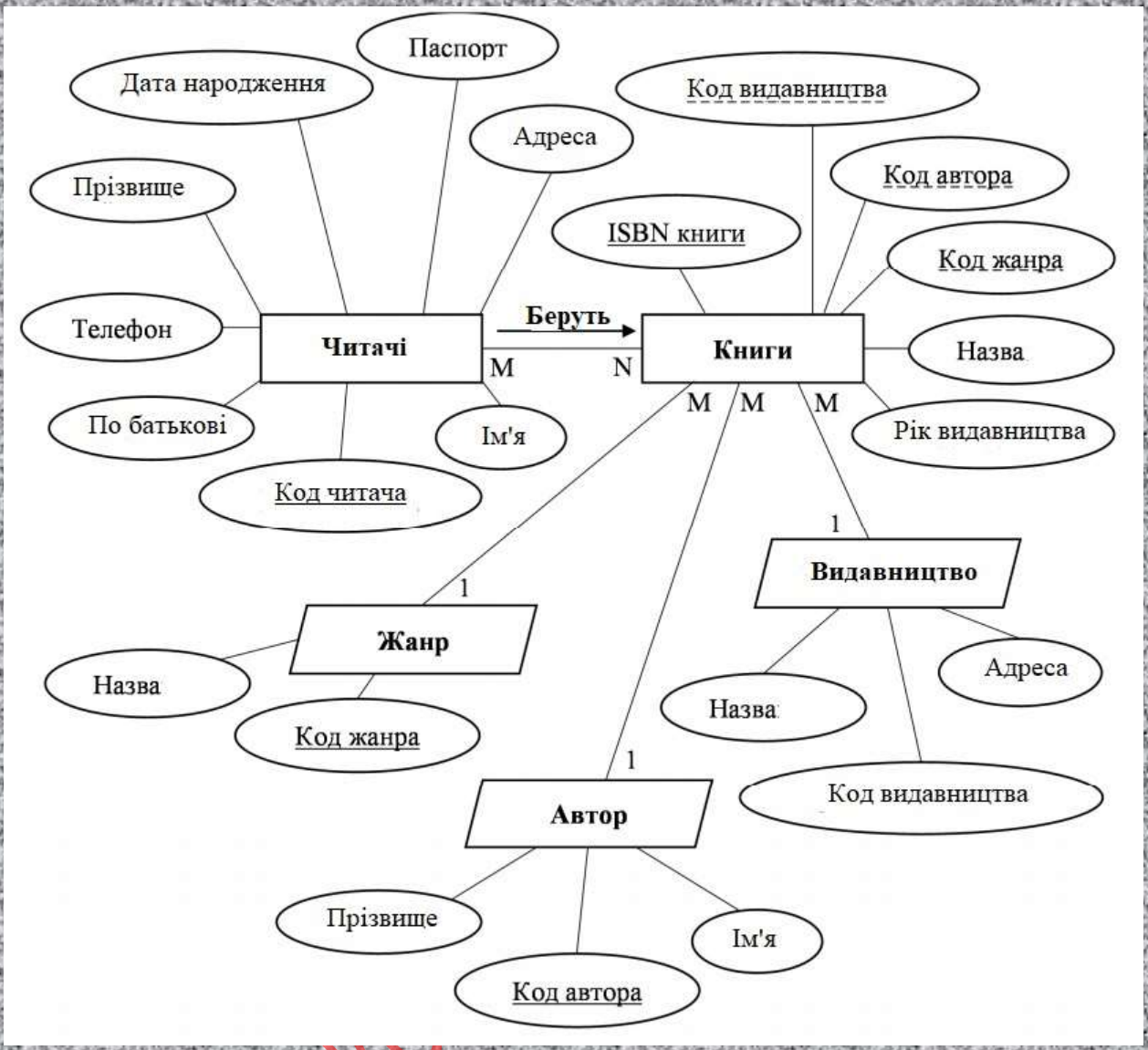


Рис. 2.7. ER-діаграма зв’язку сутностей «Книги» та «Читачі»

Зв'язок «Беруть» має розмірність M:N, тобто «багато-до-багатьох», так як одна книга може потрапити до багатьох читачів і один читач може взяти багато книг. Уявімо даний зв'язок двома зв'язками «один-до-багатьох» і асоціацією «Видані книги». До асоціації «Видані книги» також приєднується зв'язок «один-до-багатьох» з сутністю «Співробітники», тобто один співробітник може видати безліч книг, але одна книга не може бути видана багатьма співробітниками (за конкретний момент часу). Остаточний варіант ER-діаграми представлений на рис. 2.8.

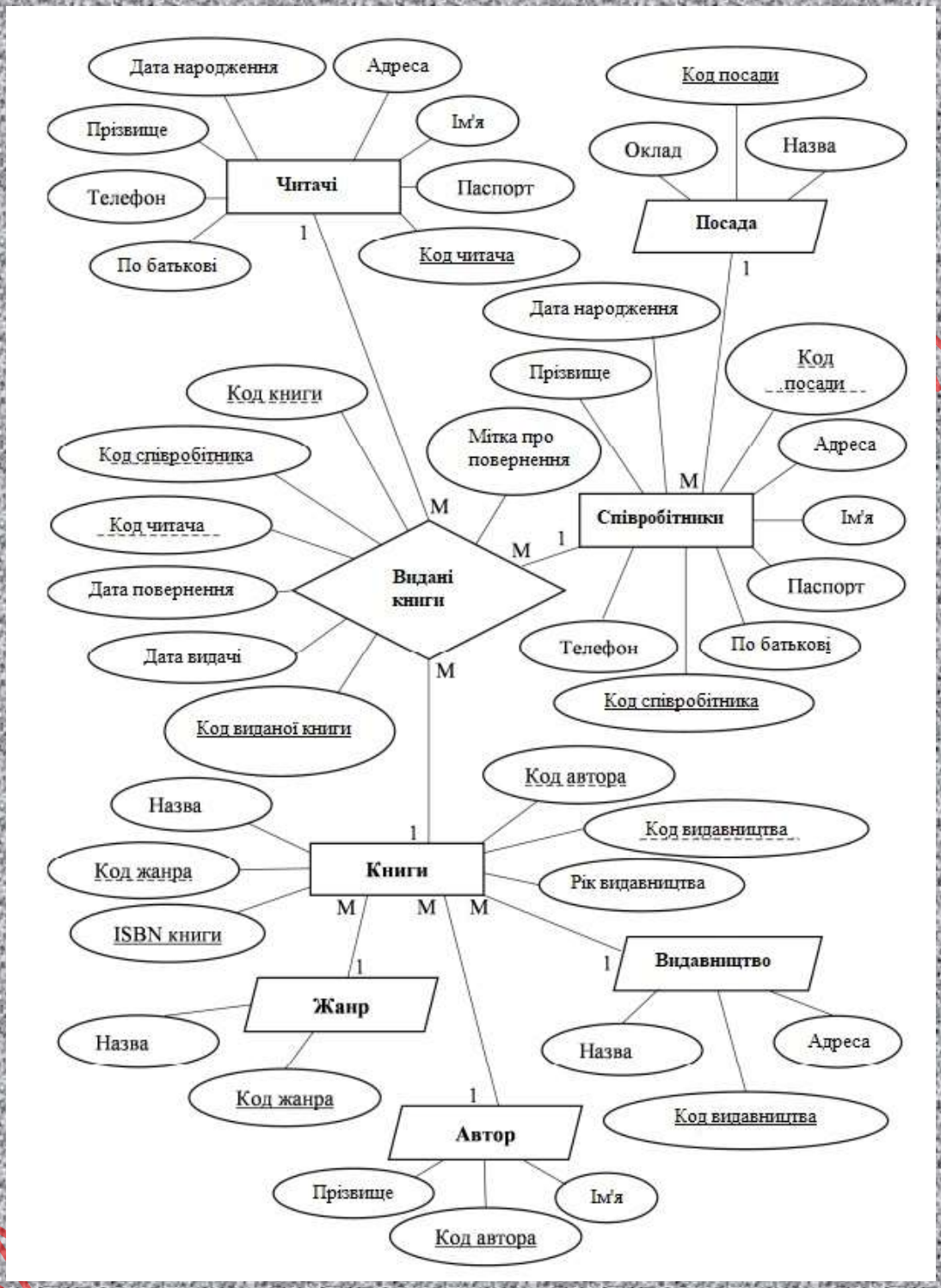


Рис. 2.8. Остаточний варіант ER-діаграми

Логічна структура бази даних, а також сама заповнена даними БД є відображенням реальної предметної області [37]. Тому на проектне рішення безпосередньо впливає специфіка, відображеної предметної області, що відображена в інфологічної моделі і виборі конкретної СУБД. По ряду причин,

платформа SQL Server Management Studio 2016 є оптимальним рішенням при виборі середовища управління базою даних.

Реляційна база даних повинна бути нормалізована. Процес нормалізації має на меті усунення надмірності даних і полягає у приведенні до третьої нормальної форми.

Перша нормальна форма (1НФ) вимагає, щоб таблиця була плоскою і не містила повторюваних груп, іншими словами вона не повинна містити осередків, що складаються з декількох значень.

Друга нормальна форма (2НФ) вимагає, щоб всі поля таблиці залежали від первинного ключа, тобто, щоб первинний ключ однозначно ідентифікував запис.

Для третьої нормальної форми (3НФ) потрібно, щоб всі не ключові стовпці таблиці залежали від первинного ключа таблиці, але були незалежними один від одного.

Для четвертої нормальної форми (4НФ) потрібно, щоб в одній таблиці не містилися незалежні елементи даних, якщо між ними існує відношення "багато-до-багатьох" [41, с.28-30].

Виходячи з вищесказаного, база даних бібліотека «Світогляд» OUTLOOK (labrary) буде являти собою вісім пов'язаних таблиць: Книги (book), Автори (author), Жанр (genre), Видавництва (publishing), Читачі (reader), Співробітники (employee), Посади (post) і Видані_книги (not_book). Наявність таблиць «Автори», «Жанр», «Видавництва» і «Посади» забезпечують усунення надмірності. Таблиця «Видані_книги» усуває відношення «багато-до-багатьох», що відповідає вимогам 4НФ.

Таблиця 2.3 містить інформацію про таблиці, що знаходяться в складі БД OUTLOOK «labrary».

Таблиця 2.3

Атрибути та відношення в складі таблиць БД OUTLOOK «labrary»

Таблиця	Ім’я стовпчика	Тип даних	Ключ таблиці по полю		Обмеження
			Первинний ключ	Зовнішній ключ	
1	2	3	4	5	6
book	book_ISBN	Int	+		identity, not null
	book_name	varchar(40)	–		not null
	author_id	Int		+	
	publish_id	Int		+	
	book_year	date	–		
	genre_id	Int		+	
author	author_id	Int	+		identity, not null
	author_name	varchar(40)	–		not null
	author_forname	varchar(25)	–		not null
genre	genre_id	int	+		identity, not null
	genre_name	varchar(30)	–		not null
publishing	publish_id	int	+		identity, not null
	publish_name	varchar(20)	–		not null
	publish_adress	varchar(50)	–		not null
reader	read_id	int	+		identity, not null
	read_name	varchar(40)	–		not null
	read_birth	date	–		not null
	read_adress	varchar(20)	–		not null
	read_num	int	–		
	read_passport	int	–		
	read_forname	varchar(25)	–		
	read_lastname	varchar(25)	–		
employee	empl_id	int	+		identity, not null
	empl_name	varchar(40)	–		not null
	empl_birth	date	–		not null
	empl_adress	varchar(20)	–		not null
	empl_num	int	–		
	empl_passport	int	–		
	post_id	int	–		
	empl_forname	varchar(25)	–		not null
	empl_lastname	varchar(25)	–		not null
post	post_id	int		+	identity, not null
	post_name	varchar(20)	–		not null
	post_salary	smallmoney	–		not null
not_book	nbook_id	int	+		identity, not null
	book_id	int	–		
	read_id	int	–		
	nbook_isdate	date	–		not null
	nbook_retdate	date	–		
	returnflag	bit	–		
	empl_id	int	–		

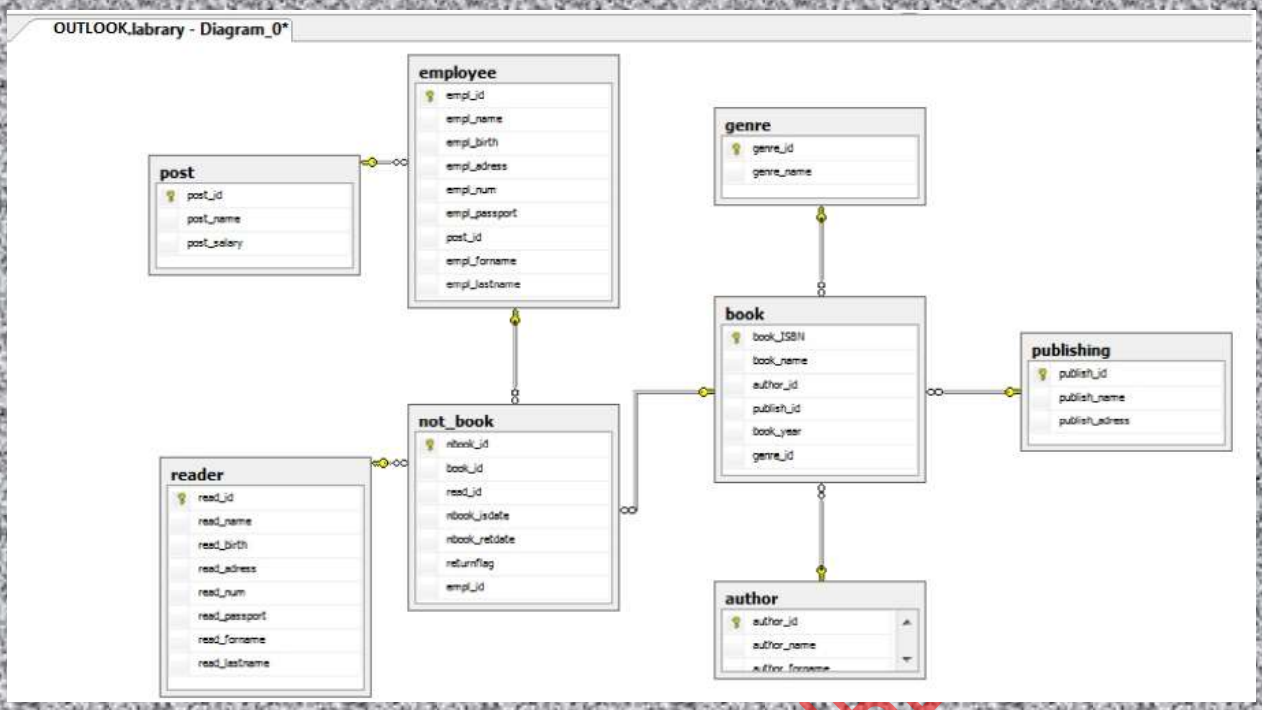


Рис. 2.9. Діаграма бази даних бібліотеки «Світогляд»

Даталогічна модель бази даних розроблена відповідно до принципів нормалізації, наступний етап розробки – створення фізичної моделі даних.

Фізичне проектування – створення бази даних в середовищі SQL Server Management Studio 2016 засобами універсальної комп'ютерної мови SQL, що застосовується для створення, модифікації та управління даними в реляційних базах даних.

1. Запит на створення бази даних

create database OUTLOOK.labrary

2. Запит на створення таблиць

create table employee (empl_id int primary key identity (1,1), empl_name varchar (40) not null, empl_birth date not null, empl_adress varchar (20) not null, empl_num int, empl_passport int not null, post_id int)

create table post (post_id int primary key identity (1,1), post_name varchar (20) not null, post_salary smallmoney not null)

create table publishing (publish_id int primary key identity (1,1), publish_name varchar (20) not null, publish_burg varchar (20) not null)

create table genre (genre_id int primary key identity (1,1), genre_name varchar (30) not null)

create table author (author_id int primary key identity (1,1), author_name varchar (40) not null)

create table book (book_id int primary key identity (1,1), book_name varchar (40) not null, author_id int, publish_id int, book_year date, genre_id int)

create table reader (read_id int primary key identity (1,1), read_name varchar (40) not null, read_birth date not null, read_adress varchar (20) not null, read_num int, read_passport int not null)

create table not_book (nbook_id int primary key identity (1,1), book_id int,read_id int, nbook_isdate date not null, nbook_retdate date, returnflag bit, empl_id int)

3. Запит на створення зовнішніх ключів

alter table employee add constraint a1 foreign key (post_id) references post (Post_id)

alter table book add constraint b1 foreign key (author_id) references author (Author_id)

alter table book add constraint c1 foreign key (publish_id) references publishing (publish_id)

alter table book add constraint d1 foreign key (genre_id) references genre (Genre_id)

alter table not_book add constraint e1 foreign key (book_id) references book (Book_ISBN)

alter table not_book add constraint g1 foreign key (read_id) references reader (Read_id)

alter table not_book add constraint k1 foreign key (empl_id) references employee (empl_id)

4. Повідомлення про порушення полів таблиць на прикладі таблиці «Publishing»

```
alter table publishing drop column publish_burg
alter table publishing add publish_adress varchar (50) not null
```

5. Запит на заповнення таблиць на прикладі таблиці «Publishing»

```
insert into publishing values ('ACCA', '61146, Харків, вул. Академіка Павлова, буд. 140-В')
insert into publishing values ('Генеза', '01133, Київ, пл. Лесі Українки, 1')
insert into publishing values ('Грамота', '04123, Київ, вул. Межова, буд. 3-А')
insert into publishing values ('КСД," '61001, Харків, вул. Хмельницького, 24')
insert into publishing values ('Наш формат', '01032, Київ, провулок Алли Горської, 5а')
insert into publishing values ('Пегас', '61057, Харків, вул. Пушкінська, 11/13')
insert into publishing values ('Ранок', '02095, Київ, вул. Княжий Затон, 4')
insert into publishing values ('Талант', 61052, 'Харків, провулок Сімферопольський, буд. 6')
```

Результат запиту продемонстрований на рисунку 2.10.

OUTLOOK .labrary - dbo.publishing

	publish_id	publish_name	publish_adress
▶	7	ACCA	61146,Харків, вул.Академіка Павлова, буд.140-В
	8	Генеза	01133, Київ, пл. Лесі Українки, 1
	9	Грамота	04123, Київ, вул. Межова, буд. 3-А
	10	КСД	61001, Харків, вул. Хмельницького, 24
	11	Наш формат	01032, Київ, провулок Алли Горської, 5а
	13	Пегас	61057, Харків, вул. Пушкінська, 11/13
	14	Ранок	02095, Київ, вул. Княжий Затон, 4
*	NULL	NULL	NULL

Рис. 2.10. Результат виконання запиту на заповнення таблиці «Publishing»

Аналогічно заповнюються інші таблиці, скріншоти яких наведені нижче.

OUTLOOK\labrary - dbo.author

author_id	author_name	author_forname
4	Сьезен	Коллінз
5	Дэн	Браун
6	Иар	Эльтеррус
7	Федор	Достоевский
8	Юсуф	Зейдан
9	Клайв	Касслер
10	Александр	Пушкин
11	Лев	Толстой
12	Бьорн	Страуступ
13	Крістофер	Паоліні
14	Уілбур	Сміт
15	Стівен	Кінг
16	Алекс	Орлов
17	Братья	Гримм
18	Александр	Войтенко
19	Дмитрий	Григорович
20	Андрей	Белянин
NULL	NULL	NULL

Рис. 2.11. Таблиця «Author»

OUTLOOK\labrary - dbo.genre

genre_id	genre_name
1	Казки
2	Наука
3	Фантастика
4	Жахи/містика
5	Пригоди
6	Детектив
7	Фентезі
8	Історична проза
9	Класика

Рис. 2.12. Таблиця «Genre»

OUTLOOK.labrary - dbo.book

book_ISBN	book_name	author_id	publish_id	book_year	genre_id
1	Разговорный английский	18	7	2019-01-20	2
3	Бременские музыканты	17	7	2017-10-12	1
4	Вкус вампира	20	8	2012-12-05	3
5	Крылья огненных драконов	16	8	2018-04-25	3
6	Таємниця синіх лісів	16	8	2018-06-14	3
7	Сезон королевской охоты	16	8	2017-05-17	3
8	Країна радості	15	9	2018-08-15	4
9	Погляд тигра	14	9	2020-04-18	5
10	Инферно	5	9	2020-07-17	6
12	Скровище Чингискана	9	9	2014-10-27	5
13	Азазель	8	9	2020-10-30	8
14	Ерагон	13	10	2018-12-27	7
15	Ерагон.Брисінг	13	10	2017-04-25	7
17	Преступление и наказание	7	13	2019-01-10	9
18	Война и мир. Том 1	11	13	2018-05-22	9
19	Евгений Онегин	10	13	2007-10-03	9
20	Стоя на краю неба	6	14	2008-02-18	7
21	Гуттаперчевый мальчик	19	14	2018-11-23	9
26	Голодные игры	4	9	2020-05-07	7
27	Мова програмування С++	12	11	2020-10-10	2
NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2.13. Таблиця «Book»

OUTLOOK.labrary - dbo.employee

empl_id	empl_name	empl_birth	empl_adress	empl_num	empl_passport	post_id	empl_forname	empl_lastname
3	Олена	1994-01-25	м. Київ	254865	НК025615	4	Байдюк	Василівна
9	Георгій	1985-10-02	м.Біла Церква	458651	МА057482	16	Мельник	Олегович
12	Олег	1992-02-15	м. Боярка	018456	ВА782452	2	Сапрун	Семенович
13	Вікторія	1995-10-27	м. Київ	201855	МВ204710	2	Пільгіна	Леонідівна
14	Марина	1999-11-01	м. Київ	025201	СП203474	3	Олексюк	Євгеніївна
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2.14. Таблиця «Employee»

OUTLOOK.labrary - dbo.not_book

nbook_id	book_id	read_id	nbook_isdate	nbook_retdate	returnflag	empl_id
2	1	7	2020-02-12	2020-06-04	True	12
3	3	5	2020-05-20	2020-08-18	True	13
4	5	8	2020-08-24	NULL	False	12
5	10	5	2020-04-01	NULL	False	12
6	18	13	2020-08-10	2020-10-17	True	13
7	27	4	2020-09-10	NULL	False	8
8	8	12	2020-10-02	2020-11-25	True	9
9	9	3	2020-09-14	NULL	False	14
10	12	1	2020-11-01	2020-11-21	True	13
11	15	11	2020-11-08	2020-11-28	True	8
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2.15. Таблиця «Not_book»

OUTLOOK.labrary - dbo.reader

read_id	read_name	read_birth	read_adress	read_num	read_passport	read_forname	read_lastname
1	Дарія	1995-02-17	м. Київ	58445	ТР524585	Бойко	Борисівна
3	Марія	1998-12-08	м. Павлоград	48420	ЕР217585	Сумська	Сергіївна
4	Сергій	1999-10-15	м. Харків	10545	НА057425	Омельченк	Андрійович
5	Семен	2000-12-28	м. Полтава	02154	ШИ760821	Зібров	Володимирович
6	Анна	1994-08-30	м. Рівне	54200	ГТ410741	Пилипчук	Дмитрівна
7	Антон	1998-11-10	м. Рівне	21545	ИО028412	Райчук	Максимович
8	Андрій	1995-07-05	м. Київ	10542	ЕП463201	Колісник	Ілліч
11	Ілля	1997-03-02	м. Львів	02124	ПР201410	Осадчий	Олегович
12	Валентина	1989-08-22	м. Київ	75100	ПМ510245	Римарчук	Петрівна
13	Софія	1987-12-14	м. Фастів	04521	ПО711452	Орищук	Леонідівна
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рис. 2.16. Таблиця «Reader»

OUTLOOK.labrary - dbo.post

post_id	post_name	post_salary
2	Бібліотекар	12000,00
3	Ведучий бібліотекар	15000,00
4	Головний бібліотекар	20000,00
16	Методист бібліотки	14000,00
NULL	NULL	NULL

Рис. 2.17. Таблиця «Post»

Наступним за розробкою фізичної моделі БД етапом є створення запитів для полегшення роботи з БД.

6. Запит на виведення списку книг на руках

select book_name as not_book, read_name, nbook_isdate from reader inner join not_book inner join book on (not_book.book_id = book.book_ISBN) on (not_book.read_id = reader.read_id) where returnflag = 'false'.

Результаты | Сообщения

	not_book	read_name	nbook_isdate
1	Крылья огненных драконов	Андрій Колісник	2020-08-24
2	Инферно	Семен Зібров	2020-04-01
3	Мова програмування С++	Сергій Омельченко	2020-09-10
4	Погляд тигра	Марія Сумська	2020-09-14

Рис. 2.18. Таблиця «Книги на руках»

7. Запит на виведення списку книг

select book_name, author_name, publish_name from author inner join book inner join publishing on (book.publish_id = publishing.publish_id) on (book.author_id = author.author_id).

Результаты | Сообщения

	book_name	author_name	publish_name
1	Разговорный английский	Александр	АССА
2	Бременские музыканты	Братья	АССА
3	Вкус вампира	Андрей	Генеза
4	Крылья огненных драконов	Алекс	Генеза
5	Таємниця синіх лісів	Алекс	Генеза
6	Сезон королевской охоты	Алекс	Генеза
7	Країна радості	Стівен	Грамота
8	Погляд тигра	Уілбур	Грамота
9	Инферно	Дэн	Грамота
10	Сокровище Чингискана	Клайв	Грамота
11	Азазель	Юсуф	Грамота
12	Ерагон	Крістофер	КСД
13	Ерагон.Брисінг	Крістофер	КСД
14	Преступление и наказание	Федор	Пегас
15	Война и мир. Том 1	Лев	Пегас
16	Евгений Онегин	Александр	Пегас
17	Стоя на краю неба	Иар	Ранок
18	Гуттаперчевый мальчик	Дмитрий	Ранок
19	Голодные игры	Сьезен	Грамота
20	Мова програмування С++	Бьорн	Наш формат

Рис. 2.19. Таблиця «Список книг»

8. Запит на виведення списку книг в наявності

select book_name, returnflag from book full outer join not_book on (Book.book_ISBN = not_book.book_id) where not (book_ISBN in (select book_id from not_book)) or (book_ISBN in (select book_id from not_book where returnflag = 1)).

Результаты | Сообщения

	book_name	returnflag
1	Разговорный английский	1
2	Бременские музыканты	1
3	Вкус вампира	NULL
4	Таємниця синіх лісів	NULL
5	Сезон королевской охоты	NULL
6	Країна радості	1
7	Сокровище Чингисхана	1
8	Азазель	NULL
9	Ерагон	NULL
10	Ерагон. Брісінг	1
11	Преступление и наказание	NULL
12	Война и мир.Том 1	1
13	Евгений Онегин	NULL
14	Стоя на краю неба	NULL
15	Гуттаперчевый мальчик	NULL
16	Голодные игры	NULL

Рис. 2.20. Таблиця «Книги в наявності»

9. Запит на виведення списку класичних книг

select book_name, genre_name from book inner join genre on (Book.genre_id = genre.genre_id) where genre_name = 'Класика'.

Результаты | Сообщения

	book_name	genre_name
1	Преступление и наказание	Класика
2	Война и мир.Том 1	Класика
3	Евгений Онегин	Класика
4	Гуттаперчевый мальчик	Класика

Рис. 2.21. Таблиця «Книги жанру класичні»

10. Запит на висновок картотеки читачів

select read_name + read_forname as read_name, read_adress, read_num from reader order by read_name asc.

Результаты | Сообщения

	read_name	read_adress	read_num
1	Андрій Колісник	м. Київ	10542
2	Анна Пилипчук	м. Рівне	54200
3	Антон Райчук	м. Рівне	21545
4	Валентина Римарчук	м. Київ	75100
5	Дарія Бойко	м. Київ	58445
6	Ілля Осадчий	м. Львів	02154
7	Марія Сумська	м. Павлоград	48420
8	Семен Зібров	м. Полтава	02124
9	Сергій Омельченко	м. Харків	10545
10	Софія Орищук	м. Фастів	04521

Рис. 2.22. Таблиця «Картотека читачів»

11. Запит на підрахунок середньої заробітної плати

```
select avg (post_salary) as avg_salary from post
```

Результаты | Сообщения

	avg_salary
1	15250,00

Рис. 2.23. Середня заробітна плата співробітника

12. Запит на підрахунок штату співробітників

```
select count (empl_id) as count_employee from employee
```

Результаты | Сообщения

	count_employee
1	5

Рис. 2.24. Штат співробітників

13. Запит на пошук читача по прізвищу

```
select read_forname, read_name, read_lastname from reader where read_forname like ('Колісник%')
```

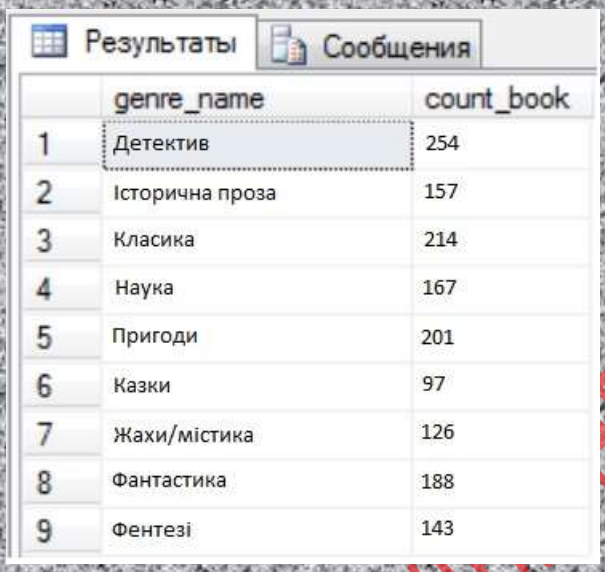
Результаты | Сообщения

	read_forname	read_name	read_lastname
1	Колісник	Андрій	Ілліч

Рис. 2.25. Результат пошуку

14. Запит на угруповання і підрахунок книг за жанрами

select genre_name, count (book_name) as count_book from genre inner join book on (genre.genre_id = book.genre_id) group by genre_name.



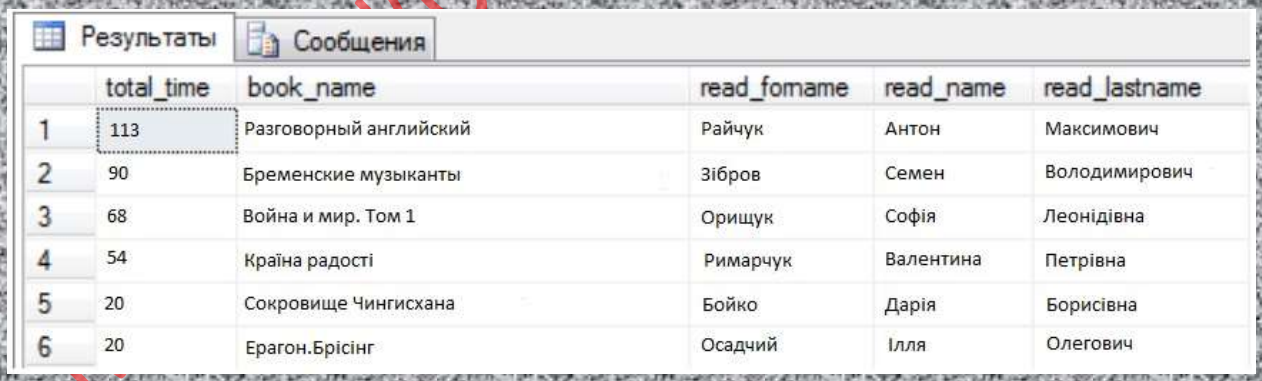
Результаты | Сообщения

	genre_name	count_book
1	Детектив	254
2	Історична проза	157
3	Класика	214
4	Наука	167
5	Пригоди	201
6	Казки	97
7	Жахи/містика	126
8	Фантастика	188
9	Фентезі	143

Рис. 2.26. Таблиця «Кількість книг по жанрам»

15. Запит на підрахунок часу, який книга була на руках

select DATEDIFF (DD, nbook_isdate, nbook_retdate) as total_time, book_name, read_forname, read_name, read_lastname from book inner join not_book inner join reader on (reader.read_id = not_book.read_id) on (Book.book_ISBN = not_book.book_id) where returnflag = 1



Результаты | Сообщения

	total_time	book_name	read_forname	read_name	read_lastname
1	113	Разговорный английский	Райчук	Антон	Максимович
2	90	Бременские музыканты	Зібров	Семен	Володимирович
3	68	Война и мир. Том 1	Орищук	Софія	Леонідівна
4	54	Країна радості	Римарчук	Валентина	Петрівна
5	20	Сокровище Чингисхана	Бойко	Дарія	Борисівна
6	20	Ерагон.Брісінг	Осадчий	Ілля	Олегович

Рис. 2.27. Таблиця «Час, який книга була на руках»

Наступний крок – створення уявлень для певних раніше груп користувачів.

16. Подання для типової групи користувачів – читачів

create view read_view as select book_name, author_name + author_forname as author_name from author inner join book full outer join not_book on (Book.book_ISBN = not_book.book_id) on (book.author_id = author.author_id) where not (book_ISBN in (select book_id from not_book)) or (book_ISBN in (select book_id from not_book where returnflag = 1)) select * from read_view.

Результаты | Сообщения

	book_name	author_name
1	Разговорный английский	Александр Войтенко
2	Бременские музыканты	Братья Гримм
3	Вкус вампира	Андрей Белянин
4	Таємниця синіх лісів	Алекс Орлов
5	Сезон королевской охоты	Алекс Орлов
6	Країна радості	Стівен Кінг
7	Сокровище Чингисхана	Клайв Касслер
8	Азазель	Юсуф Зейдан
9	Ерагон	Крістофер Паоліні
10	Ерагон.Брісінг	Крістофер Паоліні
11	Преступление и наказание	Федор Достоевский
12	Война и мир. Том 1	Лев Толстой
13	Евгений Онегин	Александр Пушкин
14	Стоя на краю неба	Иар Эльтеррус
15	Гуттаперчевый мальчик	Дмитрий Григорович
16	Голодные игры	Сьезен Коллинз

Рис. 2.28. Таблиця «Список книг в наявності з зазначенням авторів»

17. Подання для типової групи користувачів – директор

create view direct_view as select post_salary, empl_name + " " + empl_forname + " " + Empl_lastname as employee_name, post_name from employee inner join post on (employee.post_id = post.post_id) select * from direct_view

Результаты | Сообщения

	post_salary	employee_name	post_name
1	20000,00	Байдюк Олена Василівна	Головний бібліотекар
2	14000,00	Мельник Георгій Олегович	Методист бібліотеки
3	12000,00	Сапрун Олег Семенович	Бібліотекар
4	12000,00	Пільгіна Вікторія Леонідівна	Бібліотекар
5	15000,00	Олексюк Марина Євгеніївна	Ведучий бібліотекар

Рис. 2.29. Таблиця «Штат співробітників з вказаною посадою та розміром заробітної плати

Необхідність створення процедур, функцій і тригерів обумовлена спрощенням роботи співробітників. Процедура пошуку книг за вхідними даними знижує час на аналогічну процедуру в ручному виконанні. Процедура підрахунку часу до здачі книг і функція, що забезпечує оформлення замовлення на книги, також є необхідною частиною комфортної роботи оператора БД.

18. Процедура пошуку книги за вхідними даними

```
create procedure search_book @a varchar (40) as select book_name, author_name + author_forname as author_name, publish_name
from author inner join book inner join publishing on (Book.publish_id = publishing.publish_id) on (book.author_id = author.author_id)
where book_name = @a execute
search_book 'Война и мир. Том 1'.
```

Результаты | Сообщения

	book_name	author_name	publish_name
1	Война и мир. Том 1	Лев Толстой	Пегас

Рис. 2.30. Таблиця «Результат пошуку книги за вхідними даними»

19. Процедура підрахунку часу до здачі книги

```
create procedure time_notbook @a date as
declare @z date, @i int, @name varchar(40)
```

```
set @i = 1
while (@i <(select MAX (nbook_id) from not_book))
begin
        set @name = (select book_name from book where book_ISBN = @i)
        if (select returnflag from not_book where nbook_id = @i) = 0
        begin
        set @z = (select nbook_isdate from not_book where nbook_id = @i)
                if dateadd(dd,10,@z) > @a
                begin
                        declare @as char (50)
    set @as = 'до сдачи книги "' + @name + '" осталось' + convert(varchar(5),
(datediff(dd,@a,@z))) + 'дней'
                        Print @as
        end
        else
                Print'книга "'+ @name +'" задержана на ' + convert(varchar (5),
datediff(dd,dateadd(dd,10,@z),@a)) + ' дней'
        end
        set @i + = 1
end
execute time_notbook '2020-12-01'
```

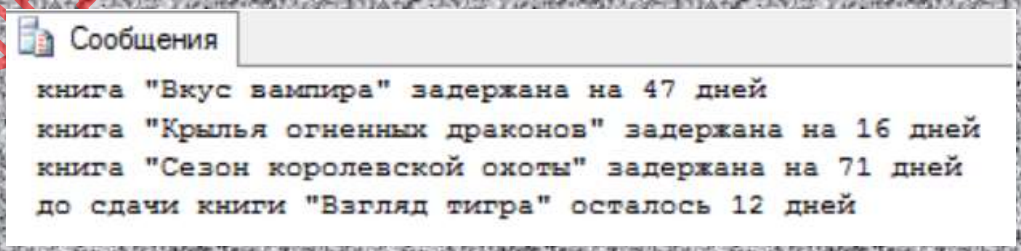


Рис. 2.31. Інформація про час до здачі книг

20. Функція для перевірки наявності книг і оформлення замовлення

```
create function order_book (@book_name varchar (40), @publish_name
```

```
varchar (40), @book_year date)
returns varchar (1000)
as begin
 declare @i int, @g varchar (1000)
 set @ =i = 1
 set @g = 'Книга' + @book_name + '' + convert (varchar(5), year (@book_year))
+ 'Года издания (издательство:' + @publish_name + ') не найдено. Оформить заказ книги. '
 while (@i <(select MAX (book_ISBN) from book))
 begin
  if (((select book_name from book where book_ISBN = @i) = @book_name) and ((select publish_name from book inner join publishing on (Book.publish_id = publishing.publish_id) where book_ISBN = @ i) = @publish_name) and ((select book_year from book where book_ISBN = @ i) = @book_year))
  begin
   set @g = 'Книга "' + @book_name + '"' + convert (varchar (5), year (@book_year)) + 'года издания (издательство:' + @Publish_name + ') имеется в библиотеке. Заказ не требуется. ' break
 end
 set @i + = 1
 end
 return @g
 end
 select dbo.order_book ( 'Разговорный английский, 'ACCA', '2019-01-20')
```

Результаты | Сообщения

	(Отсутствует имя столбца)
1	Книга Разговорный английский 2019 года издания (издательство: ACCA) не найдена. Оформите заказ книги

Рис. 2.32. Інформація про час до здачі книг

21. Тригер на видалення інформації з таблиць

```
create trigger delete_book
on not_book for delete as
delete book
from book, deleted
where book.book_ISBN = deleted.book_id
print 'Deleted from not_book'
go
```

Транзакція – це сукупність однієї або декількох SQL-інструкцій, які мають початок і кінець. В кінці транзакції відбувається або її скасування, або підтвердження. Скасування транзакції називається відкатом (rollback), так як йде послідовна скасування всіх зроблених змін. Підтвердження транзакції називається фіксацією (commit). Транзакція на зміну прапора при внесенні інформації про дату повернення книги має логічне підгрунтя, так як в разі, якщо оператор при внесенні дати повернення не внесе змін в поле мітки, виникне дезінформація, у всіх запитах дана книга буде виводиться що не наявна, що може спричинити за собою можливий збій системи.

22. Транзакція на зміну прапора при внесенні інформації про дату повернення книги

```
declare @i int
set @i = 2
update not_book set nbook_retdate = '2020-12-01' where nbook_id = @i
begin tran
if ((select nbook_retdate from not_book where nbook_id = @i) is not null)
begin
update not_book set returnflag = 'true' where nbook_id = (select nbook_id
from not_book where nbook_id = @i)
end
if (@@ error = 0) begin
```

```
commit tran
end
else begin
print 'Возникла ошибка'
rollback tran
end
```

OUTLOOK.labrary - dbo.not_book

nbook_id	book_id	read_id	nbook_jsdate	nbook_retdate	returnflag	empl_id
2	1	7	2020-02-12	2020-06-04	True	12
3	3	5	2020-05-20	2020-08-18	True	13
4	5	8	2020-08-24	NULL	False	12

Рис. 2.33. До внесення дати повернення книги (nbook_id=5)

OUTLOOK.labrary - dbo.not_book

nbook_id	book_id	read_id	nbook_jsdate	nbook_retdate	returnflag	empl_id
2	1	7	2020-02-12	2020-06-04	True	12
3	3	5	2020-05-20	2020-08-18	True	13
4	5	8	2020-08-24	2020-12-01	True	12

Рис. 2.34. Після внесення дати повернення книги (nbook_id=5)

Для спрощення управління правами користувача беруться ролі, які схожі на групи системи безпеки Windows. При роботі окремі логічно пов'язані користувачі поміщаються в одну роль. У цій базі даних присутні три групи користувачі-співробітники, читачі та директор. Ролі дозволяють обмежити дії, які користувачі можуть виконувати з елементами бази даних: таблицями, уявленнями, функціями або процедурами. Наприклад, для користувачів-читачів заборонено редагування в будь-якому вигляді, уявлення є тільки для перегляду; користувачі-співробітники мають право на читання, запис і видалення даних; користувач-директор має право на читання і оновлення записів уявлення.

23. Створення ролей і дозволів для роботи з уявленнями

```
create role employee_role;
create role reader_role;
create role director_role;
grant select, update on direct_view to director_role
```

grant select, insert, delete on empl_view to employee_role

grant select on read_view to reader_role

deny update, insert, alter, delete on read_view to reader_role.

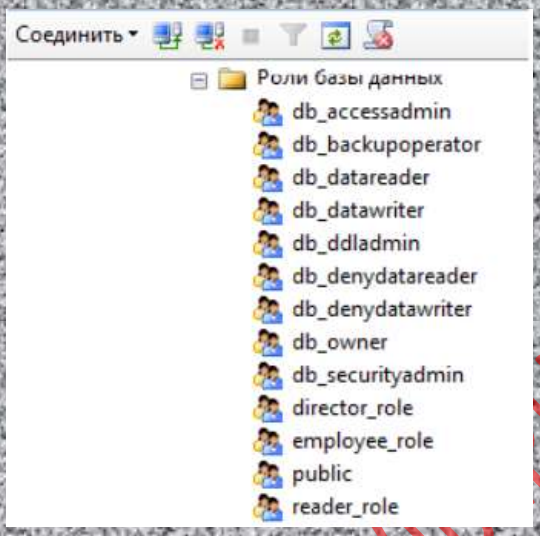


Рис. 2.35. Ролі бази даних бібліотеки «Світогляд

Висновки: сучасні програмні системи стають складнішими, завдяки чому рішення багатьох завдань в різних областях людської діяльності автоматизується, в тому числі це відбилося і на роботі бібліотеки «Світогляд».

У процесі створення магістерської роботи була розроблена реляційна база даних по предметної області бібліотеки «Світогляд».

Проектування проходило в чотири етапи:

- створення БД по технології allfusion process та data modeler;
- побудова інфологічної моделі даних;
- побудова датологочної моделі даних;
- розробка фізичної моделі даних.

Для спрощення роботи з базою, були створені ряд запитів, збережених процедур, транзакцій, функцій і тригерів, які автоматизують основні процеси взаємодії користувача з каталогами даних. Бази даних орієнтована на роботу з декількома групами користувачів, що значно розширює область її застосування.

Висновки до розділу 2

В другому розділі магістерської роботи було проаналізовано предметну область та побудовано реляційну модель бази даних бібліотеки «Світогляд» за допомогою структурної мови програмування SQL, що дало змогу отримати наступні суттєві результати та зробити такі висновки.

1. Проаналізовано предметну область бібліотеки та зокрема з'ясовано, що сучасні програмні системи стають складнішими, завдяки тому, що означає, що рішення багатьох завдань в різних областях людської діяльності автоматизується, в тому числі це відбилося і на роботі бібліотеки.

2. Створено та реалізовано базу даних ІС бібліотеки із застосуванням Allfusion Process та Data Modeler. В процесі проектування БД були представлені такі етапи: функціональне моделювання в BPwin діяльності бібліотеки в стандарті IDEF0 і експорт стрілок в сферу ERwin; створення логічної і фізичної моделей даних в ERwin на основі IDEF1x.

3. Розроблено реляційну базу даних по предметній області бібліотеки «Світогляд». Розроблений програмний продукт написано мовою структурованих запитів з використанням Microsoft SQL Server та Microsoft Management Studio, яка має широкі можливості для роботи з базами даних. З'ясовано, що для спрощеної роботи з базою, було створено ряд запитів, збережених процедур, транзакцій, функцій і тригерів, які автоматизують основні процеси взаємодії користувача з каталогами даних. Підтверджено, що створена база даних орієнтована на роботу з декількома групами користувачів, що значно розширює область її застосування.

РОЗДІЛ 3

ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

3.1. Оптимізація продуктивності в СУБД MICROSOFT SQL SERVER

У магістерській роботі була реалізована інформаційна система бібліотека «Світогляд», що використовує у якості сховища та архіву даних реляційну базу даних Microsoft SQL Server. Вибір цієї бази даних обґрунтований тим, що ця СУБД, робота якої виконується не тільки на хмарних серверах, але й також на локальних серверах, причому можна комбінувати типи застосовуваних серверів одночасно. Microsoft SQL Server є адаптованим продуктом для операційної системи Linux та Windows. Однією з унікальних особливостей версії 2016 року є temporal data support (тимчасова підтримка даних), яка дозволяє відстежувати зміни даних з плином часу. Остання версія Microsoft SQL-сервер підтримує dynamic data masking (динамічне маскування даних), яке гарантує, що тільки авторизовані користувачі можуть бачити конфіденційні дані.

Налаштування SQL-коду з точки зору оптимізації продуктивності на практиці починається уже після того, як основний функціонал БД розроблений, і велика частина логічних помилок усунена. Для оптимізації продуктивності роботи БД ефективно створити індекси, так як при правильному їх проектуванні можна підвищити продуктивність системи обробки даних. Індекси в SQL Server створюються для таблиць і уявлень у вигляді особливих упорядкованих структур на окремих сторінках бази даних. Індекс являється покажчиком на відповідний рядок таблиці або подання, в його склад може входити один або більше стовпців таблиці, яка індексується.

У багатьох випадках наявність індексів для таблиць бази даних і уявлень може поліпшити продуктивність системи, сильно скоротити час, що витрачається на вибірку і впорядкування даних. Якщо в системі часто використовуються запити, в яких умова вибірки даних в точності відповідає структурі одного зі створених індексів (або частково відповідають старшій структурі індексу), то такі запити виконуються з високою швидкістю.

Індекси являються досить потужним засобом збільшення продуктивності сервера. Індекси повинні створюватися для стовпців, які будуть часто використовуватися в параметрі WHERE або, меншою мірою, в параметрі ORDER BY. Однак, чим більше індексів в таблиці, тим повільніше будуть проводитися операції вставки. З технічної точки зору кожен додатковий індекс позначається і на обновленій наявній інформації, але в цьому випадку проблема не є настільки глобальною, оскільки змінам повинні будуть піддатися тільки ті індекси, для яких модифікується стовпець, який виступає в якості частини ключа. Таким чином, якщо база даних організована так, що велика частина часу присвячена виконанню операторів SELECT, а зміни вносяться порівняно рідко, то чим більше індексів, тим краще. В іншому випадку потрібно обмежитися індексами тільки для найбільш часто використовуваних стовпців. Велику допомогу в їх визначенні може надати майстер настройки індексів – спеціальний інструмент MS SQL Server, призначений для дослідження робочого навантаження і пошуку оптимального варіанту внесення змін до індексів.

Першим кроком в роботі, було створено збережену процедуру, що запускається по розкладу кожного дня та яка має здатність оптимізувати всі таблиці бази даних. Для MS SQL Server досить виконати REORGANIZE або REBUILD для всіх кластерних індексів, що призводить до оптимізації відповідної таблиці, на яких побудований індекс. (В випадку кластерного індексу рядка індексуються таблиці, які розташовуються в базі даних як кінцеві елементи в ієрархічному розміщенні елементів індексу).

Для початку необхідно створити агрегуючу таблицю:

CREATE TABLE [book_statistics] ([total] INTEGER NOT NULL, [given] INTEGER NOT NULL, [rest] INTEGER NOT NULL)

Далі напишемо код для тригерів на всіх трьох операціях (вставки, оновлення та видалення) для обох таблиць (book і not_book). Для таблиці «Book»:

CREATE TRIGGER [upd_bk_sts_on_book_ins] ON [book] AFTER INSERT AS UPDATE [book_statistics] SET [total] = [total] + (SELECT SUM([book_ISBN]) FROM [inserted])

UPDATE [boos_statistics] SET [rest] = [total] - [given]

GO

Для таблиці «not_book»:

CREATE TRIGGER [upd_bk_sts_on_not_book_ins] ON [not_book] AFTER INSERT AS DECLARE @delta INT = (SELECT COUNT(*) FROM [inserted] WHERE [nbook_isdate] = 'Y')

UPDATE [book_statistics] SET [rest] = [rest] - @delta, [given] = [given] + @delta

GO

Запишемо запит, який надає інформацію за всіма індексами, відповідними таблицями та полями.

SELECT [tables].[name] AS [table_name], [indexes].[name] AS [index_name], [indexes].[type] AS [index_type], [stats].[index_type_desc] AS [index_type_desc], [indexes].[object_id] AS [index_object_id], [columns].[name] AS [column_name], [stats].[avg_fragmentation_in_percent] AS [avg_fragm_perc], [stats].[avg_page_space_used_in_percent] AS [avg_space_perc]

FROM sys.indexes AS [indexes]

INNER JOIN sys.index_columns AS [index_columns] ON [indexes].[object_id] = [index_columns].[object_id] AND [indexes].[index_id] = [index_columns].[index_id]

INNER JOIN sys.columns AS [columns] ON [index_columns].[object_id] = [columns].[object_id] AND [index_columns].[column_id] = [columns].[column_id]

INNER JOIN sys.tables AS [tables] ON [indexes].[object_id] = [tables].[object_id]

INNER JOIN sys.dm_db_index_physical_stats(DB_ID(DB_NAME()), NULL, NULL, NULL, 'SAMPLED') AS [stats] ON [indexes].[object_id] = [stats].[object_id] AND [indexes].[index_id] = [stats].[index_id]

ORDER BY [tables].[name], [indexes].[name], [indexes].[index_id], [index_columns].[index_column_id]

В результаті виконання такого запиту ми отримаємо такі дані:

Таблиця 3.1

Індекси відповідним таблицям та полям

table_name	index_name	Index_type	Index_type_desc	index_object_id	column_name	avg_fragm_pec	avg_space_perc
book	PK_book	1	CLUSTERED INDEX	245575913	author_id	0	3.22465
book	PK_book	1	CLUSTERED INDEX	277576027	publish_id	0	3.22465
author	PK_author	1	CLUSTERED INDEX	309576141	author_id	0	2.59451
genre	PK_genre	1	CLUSTERED INDEX	309576141	genre_id	0	2.37212
genre	UQ_genre	2	CLUSTERED INDEX	341576255	genre_name	0	5.25628
publishing	PK_publishing	1	CLUSTERED INDEX	341576255	publishing_id	0	1.86557
reader	PK_reader	1	CLUSTERED INDEX	373576369	reader_id	0	2.48851
employee	PK_employee	1	CLUSTERED INDEX	374265652	employee_id	0	2.28564
employee	PK_employee	1	CLUSTERED INDEX	385458621	post_id	0	2.28564
post	PK_post	1	CLUSTERED INDEX	397122562	post_id	0	1.57257
not_book	PK_not_book	1	CLUSTERED INDEX	405576483	nbook_id	0	1.95206
not_book	PK_not_book	1	CLUSTERED INDEX	4154458218	book_id	0	1.95206
not_book	PK_not_book	1	CLUSTERED INDEX	437576597	read_id	0	1.95206

У збереженій процедурі нам будуть необхідні тільки три поля: ім'я таблиці, ім'я індексу та відсоток фрагментації. До того ж складені кластерні

індекси (наприклад, PK_book) повинні бути представлені тільки один раз. Спростимо запит так, щоб залишити тільки необхідні дані.

Таблиця 3.2

Короткий набір даних по всім індексам

table_name	index_name	avg_fragm_ pec
Book	PK_book	0
Author	PK_author	0
Genre	PK_genre	0
Publishing	PK_publishing	0
Reader	PK_reader	0
employee	PK_employee	0
Post	PK_post	0
not_book	PK_not_book	0

Тепер використовуємо отриманий запит в збереженій процедурі. Запустити отриману збережену процедуру можна наступним чином:

```
EXECUTE OPTIMIZE_ALL_TABLES
```

Наведемо код, за допомогою якого виконується дана операція.

```
USE msdb
GO
EXEC dbo.sp_add_job @job_name = V'Pilgina [book_statistics] update'
GO
EXEC sp_add_jobstep @job_name = V'Pilgina [books_statistics] update', @step_name = V'Execute
        UPDATE_BOOK_STATISTICS stored procedure', @subsystem = V'TSQL', @command = V'EXECUTE
        UPDATE_BOOK_STATISTICS', @database_name = OUTLOOK.labrary_ex_2020_mod'
GO
```

EXEC dbo.sp_add_schedule @schedule_name = N'UpdateBookStatistics', @freq_type = 4, @freq_interval = 4, @freq_subday_type = 8, @freq_subday_interval = 1, @active_start_time = 000105

USE msdb

GO

Якщо розглянути обрані поля результату такого запиту, можна спостерігати наступну картину. В таблиці 3.3 представлена подія, яка активує раз на годину збережену процедуру.

Таблиця 3.3

Результати запиту

Name	enabled	freq_type	freq_interval	freq_sub-day_type	freq_sub-day_inte-rval	Active_start_date	active_start_time
UpdateBookStatistics	1	4	4	8	1	20160427	000105
DailyOptimizeAllTables	1	4	4	1	1	20160427	000100

На цьому етапі оптимізації БД рішення для MS SQL Server завершено.

3.2. Постановка задачі та аналіз рівня якості варіантів реалізації функцій

Проектування програмного продукту зазвичай потребує значних трудових та фінансових витрат. Для максимально ефективного використання наявних ресурсів потрібно завчасно провести аналіз можливих варіантів створення, що дозволить обрати серед них найбільше раціональний.

Функціонально-вартісний аналіз (ФВА) — це метод дослідження функцій об'єкта (виробу, процесу, структури або програми), що базується на системному підході і спрямований на формування оптимальних співвідношень між корисністю функцій об'єкта, якістю їх реалізації і витратами на всіх етапах життєвого циклу.

Основні положення функціонально-вартісного аналізу:

1. Вивчення та вдосконалення об'єкту проводиться через призму функцій, який він виконує;

2. Резервом зниження собівартості є скорочення зайвих витрат;

3. Зайві витрати пов'язані з нераціональним використанням матеріалів, з недосконалістю виробничого процесу або конструкції виробів, із застарілою технологією виробництва, з помилковими рішеннями;

4. Завдання ФВА – досягнення функціональності об'єкту мінімальними витратами, в інтересах як виробників, так і споживачів;

5. Об'єктами ФСА можуть виступати як вироби, так і технології, організація виробництва, що комплектує елементи і матеріали.

ФВА проводиться в кілька етапів, представлених в таблиці 3.4.

Таблиця 3.4

Етапи ФВА

Етапи ФВА	**Зміст етапів**
Підготовчий	- популяризація методу - побудова організаційної структури ФСА - навчання спеціалістів ФСА сучасним методам вирішення технічних завдань - підбір об'єкта для проведення ФВА
Інформаційний	- збір і систематизація інформації розробки структурної схеми об'єкта - визначення витрат виробництва для складових частин об'єкта - вибір складових частин з високим рівнем витрат
Аналітичний	- визначення функцій складових частин об'єкта - розподіл функцій виробів - побудова таблиці зіставлення коефіцієнтів значущості функції і їх вартості - визначення можливих шляхів зниження витрат
Творчий	- генерація ідей (дивергенція, розширення меж) - трансформація отриманих ідей - конвергенція (звуження меж)
Рекомендаційний	- експертиза отриманих ідей - оцінка працездатності - оцінка економічної доцільності перевірки наявності ресурсів для здійснення ідей

	- оцінка конструкторських, технологічних і виробничих можливостей виробництва
Етап впровадження	- погодження плану - графіка впровадження рекомендацій ФВА з іншими розділами плану підвищення ефективності виробництва - контроль виконання плану-графіка впровадження - прийняття заходів щодо дотримання плану-графіка

У даній роботі проводиться оцінювання основних характеристик програмного продукту, який здійснює керування даними, транзакціями, ведення обліку виконаних операцій та забезпечення ефективної обробки великих обсягів інформації

Врахувавши економічні фактори та характеристики продукту, що впливають на продуктивність роботи та на його сумісність з апаратним забезпеченням, нижче наведено аналіз різних варіантів реалізації програмного модуля з метою вибору найбільш оптимального варіанту. Тому варто обрати систему показників якості програмного продукту.

Технічні вимоги до бази даних наступні:

- Простота в оновленні та збереженні даних;
- Забезпечення високої швидкості в обробці інформації в реальному часі;
- Забезпечення зручної роботи з користувачами та простої взаємодії з розробниками на підставі використання бази даних як модуля;
- Впровадити програмний продукт з мінімальними витратами.

Головна функція F0 – це проектування бази даних, яка дозволяє збирати та структурувати інформацію. Виходячи з конкретної мети, можна виділити наступні основні функції програмного продукту:

F1 – використання готових бібліотек

F2 – вибір системи управління базами даних

F3 – вибір інструменту для бази даних

Кожна з основних функцій може мати декілька варіантів реалізації, які зображені в таблиці 3.5.

Таблиця 3.5

Варіанти реалізації заданих функцій СУБД

Функція	**Варіанти реалізації**		
	A	**B**	**C**
F1	Написання алгоритмів вручну	Використання готових бібліотек	
F2	MYSQL	MemSQL	MS SQL SERVER
F3	HeidiSQL	SQL Server Management Studio	MemSQL Ops

Варіанти реалізації заданих функцій відображені у морфологічній карті системи (рис. 3.1). На основі цієї карти було побудовано позитивно-негативну матрицю варіантів заданих функцій (таблиця 3.6).

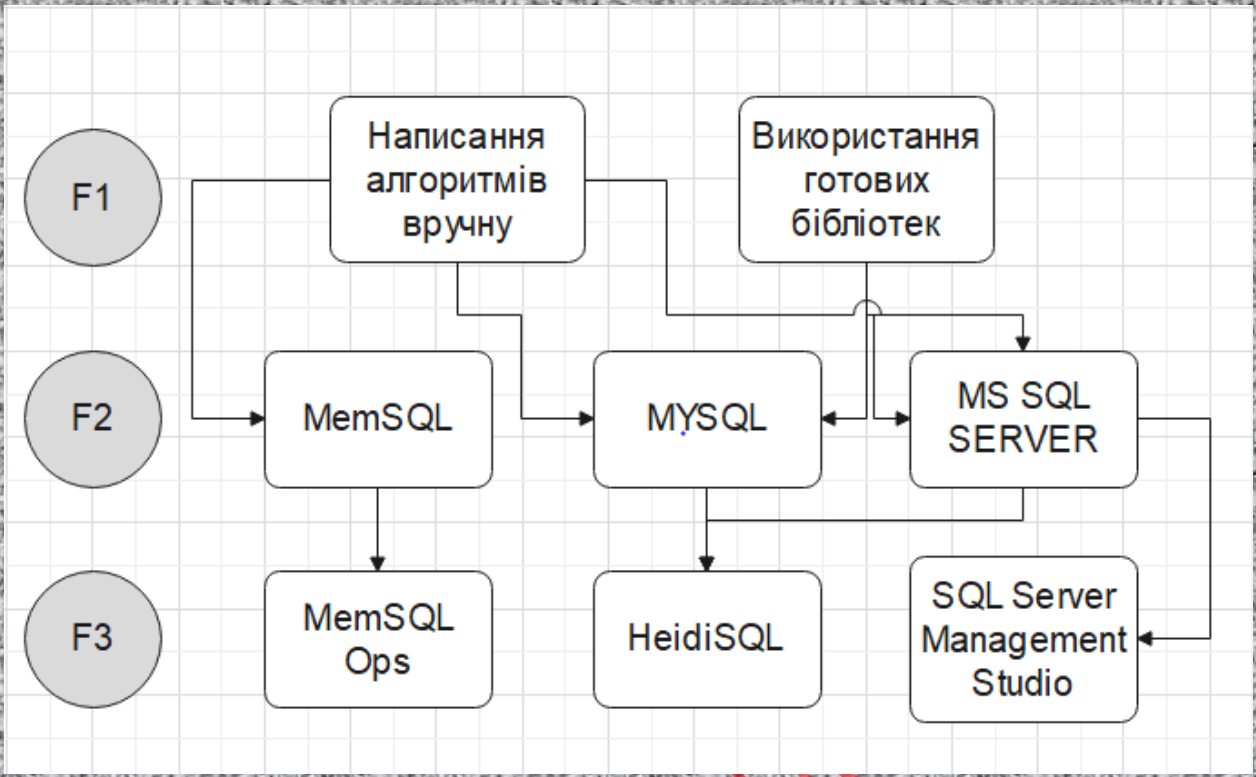


Рис. 3.1. Морфологічна карта

Морфологічна карта демонструє всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів СУБД. Для того, щоб перерахувати переваги та недоліки кожного із підходів, можна побудувати позитивно-негативну матрицю. Це матриця, у якій будуть окремо вказуватись переваги та недоліки для кожного із підходів.

Таблиця 3.6

Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	A	Повна гнучкість алгоритму	Значні витрати часу
	B	Використання готового алгоритму призведе до значної економії часу та зусиль	Менш зручний функціонал
F2	A	Мінімальна складність додатків, а також повна транзакційна підтримка та знайомий SQL і стандартні інструменти.	Не є універсальною, як традиційна система SQL та обсяг пам'яті може перевищувати кілька терабайт
	B	Пропонує багато функцій, навіть у безкоштовній версії	Файли зберігаються в базі. При великому обсязі буде

		та може працювати з іншими базами даних, включаючи DB2 і Oracle, а також максимальний об'єм на диску не має обмежень	потрібно швидкий диск (RAID, SSD)
	C	Продукт дуже простий у використанні та він дуже добре взаємодіє з іншими продуктами Microsoft, а також максимальний об'єм на диску не має обмежень	Повідомляється про проблеми з використанням служби інтеграції для імпорту файлів. Продуктивність хороша (якщо включений режим Filestream)
F3	A	Дозволяє переглядати і редагувати дані, створювати і редагувати таблиці, уявлення, процедури, тригери і заплановані події. Висока швидкість виконання операцій, безліч функцій, легке і швидке підключення до БД	Погано підходить для масових операцій, імпорту та експорту, оскільки виконання програмного коду займає багато часу
	B	Багатоплатформовий інструмент з відкритим кодом для настільних систем, призначений для роботи в хмарних, локальних або гібридних середовищах	Працює на даний момент тільки з MS SQL Server
	C	Надає простий для розуміння, інтуїтивно зрозумілий інтерфейс для користувачів різного рівня підготовки	Робочі навантаження, які створюють значний мережевий трафік в написанні коду можуть бути обмежені кластерами з меншою кількістю вузлів

На основі аналізу позитивно-негативної матриці (таблиця 3.6) можна зробити висновок, що при виборі програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони мають суттєві недоліки.

Функція F1:

Через те, що час та об’єм написання програмного коду, наявність спеціальних бібліотек та алгоритмів для обробки великого масиву даних є більш вагомим фактором, ніж тільки швидкість виконання, тому варіант B можна не розглядати.

Функція F2:

Для створення та проектування бібліотеки «Світогляд» було обрано найбільш відповідний варіант – MS SQL Server та MYSQL оскільки ці системи

управління реляційними базами даних враховують всі сучасні вимоги по роботі з даними різних форматів і з різноманітних джерел і стає природним вибором для побудови платформи інтеграції, управління та аналізу будь-яких даних.

Функція F3:

MS SQL Management Studio – один з найкращих інструментів для роботи з SQL Server. У нього єдиний мінус – це те, що він працює поки що тільки з SQL Server. Але це, мабуть, найкраще безкоштовне середовище, яке має дуже широким діапазоном можливостей для розробників. Також HeidiSQL являється безкоштовним інструментом адміністрування з відкритим вихідним кодом для MySQL і його форків, а також Microsoft SQL Server.

Таким чином будемо розглядати наступні варіанти реалізації СУБД:

F1a-F2c-F3b

F1a-F2a-F3a

Для оцінки якості програмного продукту вище вказаних та розглянутих функцій була обрана система параметрів, яка описана в наступному пункті магістерської роботи.

На підставі даних про основні функції, що повинна мати СУБД, вимог до неї, визначаються основні параметри програми, що будуть використані для розрахунку коефіцієнта технічного рівня. Для того, щоб схарактеризувати програмний продукт, будемо використовувати наступні параметри:

– X1 – Обсяг пам’яті для збереження даних;

– X2 – Час обробки даних алгоритмом;

– X3 – Потенційний обсяг програмного коду;

– X4 – Розмір бази даних.

X1: Відбиває об’єм пам’яті в оперативній пам’яті ПК, необхідний для збереження та обробки даних під час виконання програми.

X2: Відбиває час, який витрачається на дії.

X3: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

X4: Показує розмір заповненої бази даних.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 3.7.

Таблиця 3.7

Основні параметри MS SQL SERVER

Назва параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			Гірші	Середні	Кращі
Обсяг пам'яті для збереження даних	X1	МБ	32	16	8
Час обробки даних алгоритмом	X2	Мс	800	420	60
Потенційний обсяг програмного коду	X3	Кількість рядків коду	2000	1500	1000
Розмір бази даних	X4	МБ	300	150	50

За даними таблиці 3.7 будуються графічні характеристики параметрів – рис. 3.2 – рис. 3.5.

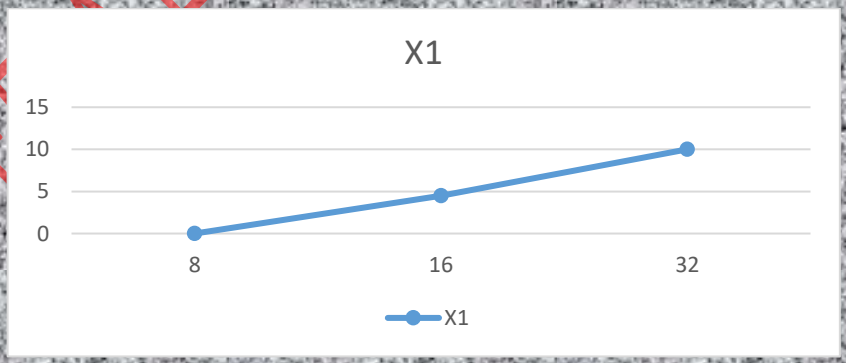


Рис. 3.2. X1, об'єм пам'яті для збереження даних

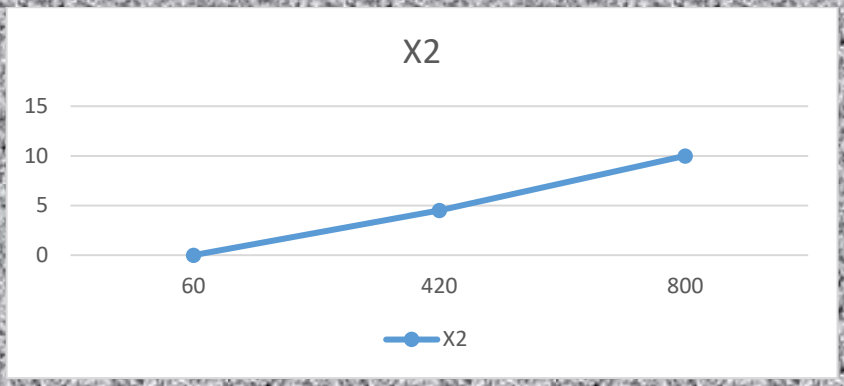


Рис. 3.3. X2, час обробки даних алгоритмом

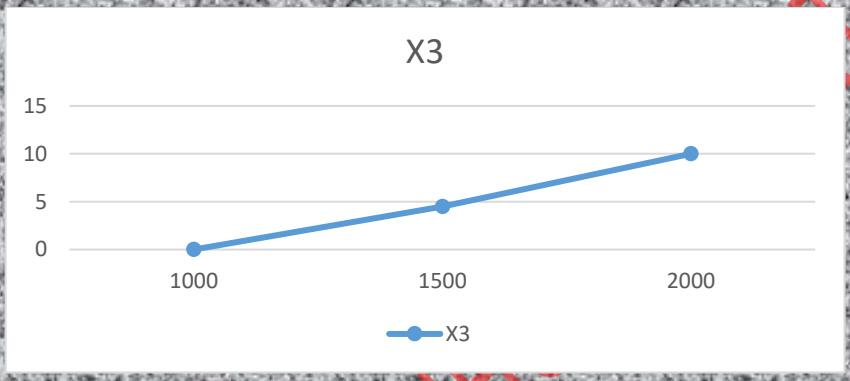


Рисунок 3.4. X3, потенційний об'єм програмного коду

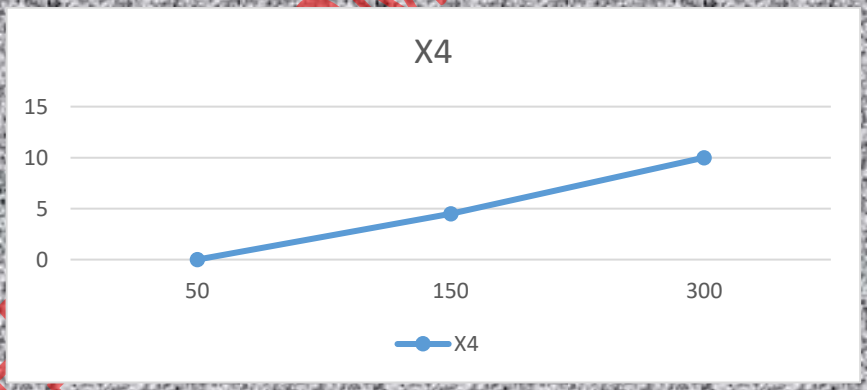


Рисунок 3.5. X4, Розмір БД, Мб

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень. Значущість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 5 людей. Визначення

коефіцієнтів значущості передбачає:

• визначення рівня значущості параметра шляхом присвоєння різних рангів;

• перевірку придатності експертних оцінок для подальшого використання;

• визначення оцінки попарного пріоритету параметрів;

• обробку результатів та визначення коефіцієнту значущості. Результати експертного ранжування наведені у таблиці 3.8.

Таблиця 3.8

Результати ранжування параметрів

Позначення параметра	**Ранг параметра за оцінкою експерта**					**Сума рангів R_i**	**Відхилення Δ_i**	**Δ_i^2**
	1	**2**	**3**	**4**	**5**			
X1	7	6	4	5	4	26	-7,75	60,06
X2	4	4	3	2	3	16	17,75	315,06
X3	8	7	9	8	8	40	6,25	39,06
X4	7	8	8	8	6	37	3,25	10,56
Разом	27	27	27	27	27	135	0	424,74

Для перевірки степені правдивості експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 135 \qquad (3.1)$$

де N – число експертів,

n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 33{,}75$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T \quad (3.2)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г) загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 424,74 \quad (3.3)$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 424,74}{5^2(4^3 - 4)} = 1,3 > W_k = 0.67 \quad (3.4)$$

Ранжування можна вважати правдивим, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати заносимо у таблицю 3.9.

Таблиця 3.9

Попарне порівняння параметрів

Параметри	Експерти					Кінцева оцінка	Числове значення
	1	2	3	4	5		
X1 і X2	>	>	>	>	>	>	1.5
X1 і X3	<	<	<	<	<	<	0.5
X1 і X4	=	<	<	<	<	<	0.5

X2 і X3	<	<	<	<	<	<	0.5
X2 і X4	<	<	<	<	<	<	0.5
X3 і X4	>	<	>	=	>	>	1.5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j, \\ 0.5 \text{ при } X_i < X_j \end{cases} \quad (3.5)$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{вi} = \frac{b_i}{\sum_{i=1}^{n} b_i} \quad (3.6)$$

де $b_i = \sum_{j=1}^{n} a_{ij}$.

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{вi} = \frac{b'_i}{\sum_{i=1}^{n} b'_i} \quad (3.7)$$

де $b'_i = \sum_{j=1}^{n} a_{ij} b_j$.

Як видно з таблиці 3.10, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 3.10

Розрахунок вагомості параметрів

Параметри x_i	**Параметри** x_j				**Перша ітерація**		**Друга ітерація**		**Третя ітерація**	
	1	2	3	4	b_i	$K_{вi}$	b_i^1	$K_{вi}^1$	b_i^2	$K_{вi}^2$
X1	1,0	1,5	0,5	0,5	3.5	0.218	19.75	0.287	70.875	0.276
X2	0,5	1,0	0,5	0,5	2.5	0.156	22.75	0.329	92.125	0.360
X3	1,5	1,5	1,0	1,5	5.5	0.344	10.75	0.156	39.875	0.156
X4	1,5	1,5	0,5	1,0	4.5	0.282	15.75	0.228	53.125	0.208
Всього:					16	1	69	1	256	1

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X1* (об'єм пам'яті для збереження даних) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X2 (*час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 800 мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 3.11):

$$K_K(j) = \sum_{i=1}^{n} K_{вi} B_{ij} \qquad (3.8)$$

де n – кількість параметрів;

$K_{вi}$ – коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 3.11

Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	A	X1	15	5	0.344	1,72
F2	A	X4	123	2.5	0.208	0.52
		X2	420	5	0.360	1,81
	C	X4	96,5	1.5	0.208	2.81
		X2	350	4	0.360	1,44
F3	B	X3	1100	1	0.156	0,16

За даними з таблиці 3.11 за формулою визначаємо рівень якість кожного з варіантів:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}]. \quad (3.9)$$

K_{K1}=1.72 + 0.52 + 1.81 + 0.16 = 4.21

K_{K2}= 1.72 + 2.81 + 1.44 + 0.16 = 6.13

Як видно з розрахунків, кращим є другий варіант, для якого коефіцієнт технічного рівня має найбільше значення.

3.3. Економічний аналіз та вибір кращого варіанта програмного продукту техніко-економічного рівня

Для розрахунку трудомісткості програмного продукту необхідно визначити [14]:

а) кількість макетів (набори даних вхідної інформації);

б) кількість різновидів форм вихідної інформації (форм друкарських документів і інформації, яка переноситься на машинні носії);

в) ступінь новизни групи задач (задачі). По цьому критерію задачі діляться на 4 групи: А – задачі, які передбачають використання принципово нових методів розробки, проведення науково-дослідних робіт; Б - розробка типових проектних рішень, оригінальних задач і систем, які не мають аналогів; В - прив'язка типових практичних рішень при умові їх зміни; розроблення задач, які мають аналогічні рішення; Г - прив'язка типових проектних рішень без їх зміни; розробка задач, які мають аналогічні рішення;

г) складність алгоритмів. Виділяють три складності: 1) алгоритми оптимізації і моделювання систем і об'єктів; 2) алгоритми обліку, звітності, статистики, пошуку; 3) алгоритми, які реалізують стандартні методи рішень, а також не передбачають використання складних чисельних і логічних методів.

д) складність організації контролю вхідної і вихідної інформації, яка характеризується наступними групами: 11 – вхідні дані і документи різноманітного розміру і структури, контроль здійснюється перехресне, тобто враховується зв'язок між показниками різних документів; 12 – вхідні дані і документи одноманітної форми і змісту, здійснюється формальний контроль; 21 – друкування документів складної багаторівневої структури, різноманітної форми і змісту; 22 – друкування документів одноманітної форми і змісту, вивід масивів даних на машинному носії.

е) вид використаної інформації, яка може бути: перемінною (ПІ); нормативно-довідковою (НДІ); банком даних (БД);

ж) мову програмування;

з) можливість використання стандартних модулів та пакетів прикладних програм.

Для визначення вартості розробки програмного продукту спочатку проведемо розрахунок трудомісткості для чотирьох завдань. Два варіанта включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Проектування бази даних;

Варіант 1 має завдання:

3. Обробка інтерфейсу готових бібліотек;

Варіант 2 має завдання:

4. Впровадження бази даних

Завдання 1 за ступенем новизни складності задач відноситься до групи В, завдання 2 – до групи Б, завдання 3 – Г, завдання 4 – Г. За видом використаної інформації які використовуються в завданнях 1, 2 та 4 відносяться до нормативно-довідкової інформації, а для завдання 3 – перемінної інформації. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1, в завданні 2 – до групи 2, в завданнях 3 та 4 – до групи 3. За складністю організації контролю вхідної і вихідної інформації всі завдання відносяться до групи 12 та 22.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_М \cdot K_{СТ} \cdot K_{СТ.М} \quad (3.10)$$

де T_P – трудомісткість розробки програмного продукту;

$K_П$ – поправочний коефіцієнт;

$K_{СК}$ – коефіцієнт на складність вхідної інформації;

$K_М$ – коефіцієнт рівня мови програмування;

$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру ступеню новизни В та групи складності алгоритму 1, трудомісткість складає:

T_P = 19 людино-днів.

Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання:

$К_П$ = 0,81.

Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх завдань рівний 1:

$К_{СК}$ = 1.

Оскільки при розробці всіх завдань використовуються стандартні модулі, то коефіцієнт у всіх варіантах буде приймати однакове значення:

$К_{СТ}$ = 0,8.

Тоді загальна трудомісткість програмування першого завдання дорівнює:

$Т_1$ = 19 · 0,8 · 1 · 0,8 = 12,31 людино-днів.

Проведемо аналогічні розрахунки для подальших досліджень.

Для другого завдання (використовується алгоритм другої групи складності, ступінь новизни Б), тобто

$Т_Р$ =27 людино-днів;

$К_П$ = 1,08;

$К_{СК}$ = 1;

$К_{СТ}$ = 0,8;

$Т_2$ = 27 · 1,08 · 1· 0,8 = 23,33 людино-днів.

Для третього завдання (використовується алгоритм третьої групи складності, ступінь новизни Г), тобто

$Т_Р$ = 8 людино-днів;

$К_П$ = 0,6;

$К_{СК}$ = 1;

$К_{СТ}$ = 0,8;

$Т_3$ = 8 · 0,6 · 1· 0,8 = 3,84 людино-днів.

Для четвертого завдання (використовується алгоритм третьої групи складності, ступінь новизни Г), тобто

$Т_Р$ = 8 людино-днів;

$К_П$ = 0,36;

$К_{СК}$ = 1;

$К_{СТ}$ = 0,8;

$Т_4$ = 8 · 0,36 · 1 · 0,8 = 2,30 людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$Т_I$ = (12,31 + 23,33 + 3,84) · 8 = 315,84 людино-годин;

$Т_{II}$ = (12,31 + 23,33 + 2,3) · 8 = 303,52 людино-годин;

Найбільш високу трудомісткість має варіант I.

В розробці беруть участь програміст та веб розробник з окладом 20000 грн. кожен, а також бізнес-аналітик з окладом 15000 грн.

Визначимо зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} \text{ грн.}$$

де М – місячний оклад працівників;

T_m – кількість робочих днів в тиждень;

t – кількість робочих годин в день.

$$С_ч = \frac{20000 + 20000 + 15000}{3 \cdot 21 \cdot 8} = 109{,}13 \text{ грн.}$$

Розраховуємо заробітну плату за формулою

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д,$$

де $С_ч$– величина погодинної оплати праці для робітника;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 109,13 · 315,84 · 1.2 = 41 361,14 грн.

II. $С_{ЗП}$ = 109,13 · 303,52 · 1.2 = 39 747,77 грн.

Відрахування на єдиний соціальний внесок в залежності від групи професійного ризику (II клас) становить 22%:

I. $С_{ВІД}$ = $С_{ЗП}$ · 0,22 = 41 361,14 · 0,22 = 9 099,45 грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0,22 = 39\,747,77 \cdot 0,22 = 8\,744,51$ грн.

Тепер визначимо витрати на оплату однієї машино-години ($С_М$)

Так як ЕОМ обслуговує одного розробника з окладом 20000 грн., з коефіцієнтом зайнятості 0,2 то для цієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 20000 \cdot 0,2 = 48\,000$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 48000 \cdot (1 + 0,2) = 57\,600$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0,22 = 57\,600 \cdot 0,22 = 12\,672$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 15 000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1,15 \cdot 0,25 \cdot 15\,000 = 4\,312,5$ грн.

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1,15 \cdot 15\,000 \cdot 0,05 = 862,5$ грн.

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0,9 = 1706,4$ годин,

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,156 \cdot 0,85 \cdot 2,5242 = 571,14$ грн.,

де N_C – середньо-споживча потужність приладу;

$K_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$C_Н = Ц_{ПР} \cdot 0.67 = 13000 \cdot 0,67 = 8\ 710$ грн.

Тоді, річні експлуатаційні витрати будуть:

$C_{ЕКС} = C_{ЗП} + C_{ВІД} + C_А + C_Р + C_{ЕЛ} + C_Н$

$C_{ЕКС} = 57\ 600 + 12\ 672 + 4\ 312,5 + 862,5 + 571,14 + 8\ 710 = 84\ 728,14$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$C_{М\text{-}Г} = C_{ЕКС} / T_{ЕФ} = 84\ 728,14 / 1706,4 = 49,65$ грн/час.

Оскільки в даному випадку всі роботи, які пов'язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$C_М = C_{М\text{-}Г} \cdot T$

I. $C_М = 49,65 \cdot 315,84 = 15\ 681,46$ грн.

II. $C_М = 49,65 \cdot 303,52 = 15\ 069,77$ грн.

Накладні витрати складають 67% від заробітної плати:

$C_Н = C_{ЗП} \cdot 0,67$

I. $C_Н = 41361,14 \cdot 0,67 = 27\ 711,96$ грн.

II. $C_Н = 39747,77 \cdot 0,67 = 26\ 631,00$ грн.

Отже, вартість розробки ПП становить:

$C_{ПП} = C_{ЗП} + C_{ВІД} + C_М + C_Н$

I. $C_{ПП} = 41361,14 + 24618 + 9099,45 + 27711,96 = 102\ 790,55$ грн.

II. $C_{ПП} = 39747,77 + 24968 + 8744,51 + 26631,00 = 100\ 091,28$ грн.

Під кінець виберемо кращий варіант техніко-економічного рівня.

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$K_{ТЕРj} = K_{Кj} / C_{Фj}, \qquad (3.11)$$

$K_{ТЕР1} = 4,21/102790,55 = 0,4 \cdot 10^{-4}$,

$K_{ТЕР2} = 6,13/100091,28 = 0,6 \cdot 10^{-4}$.

Як бачимо, найбільш ефективним є другий варіант реалізації програми з коефіцієнтом техніко-економічного рівня $K_{ТЕР2} = 0,6 \cdot 10^{-4}$.

Висновки до розділу 3

В третьому розділі магістерської роботи було оптимізовано продуктивність робити СУБД Microsoft SQL Server та проведено технічний та економічний аналіз варіантів розробки програмного продукту, що дало змогу отримати наступні суттєві результати та зробити такі висновки.

1. Оптимізовано продуктивність робити СУБД Microsoft SQL Server та досліджено, що для оптимізації продуктивності роботи БД ефективним було створення індексів, так як при правильному їх проектуванні можна підвищити продуктивність системи обробки даних.

2. Визначено основні функції програмного продукту та сформовано множинну варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій програмного продукту.

3. Досліджено альтернативні варіанти реалізації за коефіцієнтом ефективності, для обчислення яких були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

ВИСНОВКИ

Результатом виконання магістерської роботи стала розробка реляційної моделі бази даних для бібліотеки «Світогляд». В результаті проведеного дослідження на тему «Розробка бази даних для організації обліку на підприємстві» були отримані наступні висновки:

1. Проаналізовано теоретичні питання дослідження понять база даних та системи управління базами даних. Бази даних стали основою інформаційних систем і докорінно змінили методи роботи багатьох організацій. Зокрема встановлено, що в останні роки розвиток технології баз даних привів до створення досить потужних і зручних в експлуатації систем. Завдяки цьому системи баз даних стали доступними широкому колу користувачів. Крім того, SQL Server підтримує реляційну модель даних і виконує функції створення об'єктів БД (таблиць, індексів, уявлень і т.д.), здійснює перевірку цілісності БД і відповідає за безпеку даних в системі.

2. Охарактеризована методологія трьох рівнів проектування БД та їх особливості для створення реляційної бази даних бібліотеки «Світогляд». Перший рівень – інфологічне моделювання БД, яке забезпечує найбільш природні для людини способи збору і представлення інформації, що ставить собі за мету створення концептуальної моделі, що відбиває основні сутності предметної області, їх атрибути та зв'язку між сутностями. Другий рівень – даталогічне моделювання БД, деталізує інфологічну модель, перетворюючи її в логічну схему, на якій раніше виявлені суті, атрибути та зв'язку оформляються згідно з правилами моделювання для обраного виду бази даних. Третій рівень – фізичне моделювання БД, що продовжує деталізацію і дозволяє створити фізичну схему, на якій максимально враховуються технічні особливості роботи конкретної СУБД і її можливості з організації та управління структурами, розроблюваної бази даних і даними в ній.

3. Розроблена база даних для бібліотеки «Світогляд» за допомогою СУБД Microsoft SQL Server та мови запитів SQL, що надає користувачу повну автоматизацію всіх процесів роботи даної установи. А також варто зазначити, що створена база дозволяє значно оптимізувати роботу користувачів в даній організації та відповідно знизити трудовитрати співробітників. В базі даних були реалізовані запити на добавлення даних у таблицю, обновлення даних, видалення та вибір інформації з таблиць. Всі запити працюють надійно та видають чітку інформацію. Для зручності роботи з базою, було створено ряд збережених процедур, транзакцій, функцій і тригерів, які автоматизують основні процеси взаємодії користувача з каталогами даних.

4. Проведено оптимізацію продуктивності в БД за допомогою створення індексів, так як при правильному їх проектуванні можна поліпшити продуктивність системи, значно скоротити час, що витрачається на вибірку і впорядкування даних. При аналізі було встановлено, що індекси в MICROSOFT SQL Server створюються для таблиць і уявлень у вигляді особливих упорядкованих структур на окремих сторінках бази даних. Досліджено, що індекс являється покажчиком на відповідний рядок таблиці або подання, в його склад може входити один або більше стовпців таблиці, яка індексується.

5. Досліджено повний функціонально-вартісний аналіз програмного продукту в рамках магістерської роботи. Процес аналізу був виконаний в 2 етапи. Перший етап – технічний, в якому був здійснений аналіз основних функцій програмного продукту та сформовано множину варіантів їх реалізації. Другий етап – економічний, в якому був здійснений аналіз всіх можливих варіантів реалізації за допомогою коефіцієнтів ефективності та було виконане обчислення всіх заданих параметрів для вибору найефективнішого варіанта реалізації програми.

6. Проаналізовано розроблений програмний комплекс та обрано оптимальний другий варіант реалізації програмного продукту. У нього

виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР2}$ = $0{,}6 \cdot 10^{-4}$. Цей варіант реалізації програмного продукту має такі параметри: використання готових бібліотек; вибір СУБД – MICROSOFT SQL Server; вибір інструмента – SQL Management Studio. Отже, даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, непоганий функціонал та швидкодію.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. В.В. Фаронов “С 6: Учебный Курс”, Москва, Knowledge, 2001. – 245 с.

2. М.В. Сухарев “Основы C#”, издательство “Наука и техника”, 2003. – 184 с.

3. Г.В. Галисеев, “Компоненты в Delphi 6: профессиональная работа”, издательство “Диалектика”, 2004. – 325 с.

4. В.И. Ключко “Методическое указание к выполнению курсовой работы”, Краснодар, Издательство КубГТУ, 2015. – 210 с.

5. Т.М. Карпова “Базы данных: модели, разработка, реализация”, издательство “Питер”, 2001. – 212 с.

6. Кандзюба С. П. “Delphi 6/7. Базы данных и приложения. Лекции и упражнения”. – СПб.: ООО «ДиаСофтЮП», 2002. – 576 с.

7. Маклаков С.В. Моделирование бизнес-процессов с BPwin. — М.: ДИАЛОГ — МИФИ, 2002. — 224 с.

8. Саликов В.А. Проектирование информационных систем по технологии Rational Rose. Учебное пособие для дипломников. Издатель: LAP LAMBERT Academic Publishing, Saarbrücken, Deutschland, 2016. – 128 с.

9. Salikov V.A. Analysis and Specification of Requirement for Information Systems with Power Designer // Eastern European Scientific Journal (Gesellschaftswissenschaften): Düsseldorf (Germany): Auris Verlag, 2016. – 118 p.

10. Саликов В.А., студ. Гриценко А.С. Разработка модели состава данных по требованиям заказчика на основе функционального моделирования — «International scientific journal»/ Сборник научных работ VII-ой Международной научно-практической конференции: Актуальные проблемы современной науки, «28» апреля, том 2 — Киев, 2016, с. 7–12.

11. Воройский Ф.С. Основы проектирования автоматизированных библиотечно-информационных систем. М.: ГПНТБ России, 2002. – 389 с.

12. Волошина В.Н. Организация баз данных: учеб. пособие. – Владивосток: Изд-во ДВФУ, 2011. – 503 с.

13. Дубровская А.О. Информатика: Базы данных: учеб. пособие для вузов. – Владивосток: Изд-во ТГЭУ, 2009. – 104 с.

14. Кабанов В.А. Практикум Access [Электронный ресурс] / В.А. Кабанов. – М.: Инфра-М. Кузин А.В. Базы данных: учеб. пособие. – М.: Академия, 2008. – 315 с.

15. Шустова Л.И. Базы данных: учебник / Л.И. Шустова, О.В. Тараканов. – М.: НИЦ ИНФРА-М, 2016. – 336 с.

16. Брешенков А.В. Проектирование объектов баз данных в среде Access [Электронный ресурс]: учеб. пособие / А.В. Брешенков, А.М. Губарь. – Электрон. дан. – М.: МГТУ им. Н.Э. Баумана (Московский государственный технический университет имени Н.Э. Баумана), 2006. – 183 с.

17. Голенищев Э.П. Информационное обеспечение систем управления. Уч. пособие для вузов. – Ростов-н/Д.: Феникс, 2010. – 315 с.

18. Дейт К.Дж. Введение в системы баз банных / К.Дж. Дейт. – М.: Вильямс, 2001. – 1072 с.

19. Муравьев А.И. Базы данных [Электронный ресурс]: учеб. пособие. – Электрон. дан. – М.: ТУСУР (Томский государственный университет систем управления и радиоэлектроники), 2006. – 137 с.

20. Озерова Г.П. Проектирование и реализация реляционной базы данных: метод. указания для выполнения курсовых и лабораторных работ / Дальневосточный государственный технический университет; [сост. Г.П. Озерова]. – Владивосток: Изд-во Дальневосточного технического университета, 2004. – 48 с.

21. Архипенков, С. Лекции по управлению программными проектами.

22. Бродовский А.И., Воройский Ф.С., Гончаров А.М., Шрайберг Я.Л. Библиотека и информатизация. – Казань, 2016. – 229 с.

23. Вендров, А. М. Проектирование программного обеспечения экономических информационных систем; учебник.

24. Вендров, А. М. Практикум по проектированию программного обеспечения экономических информационных систем: учебное пособие.

25. Горбунов-Посадов М. М., Ермаков А. В., Луховицкая Э. С., Скорнякова Р.Ю. О выборе автоматизированной информационной библиотечной системы для библиотеки ИПМ. Препринт ИПМ № 2, М.: ИПМ, 2016. — 32 с.

26. Горевая, М.И., Клочков, Г.А., Курчеева, Г.И. Экономическая эффективность проектных решений: учебное пособие по дипломному проектированию.

27. Коннолли, Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли. - М.: Вильямс И.Д., 2017. - 1440 с.

28. Лаврик О. Л., Юдина И. Г. Использование новейших технологий для реализации информационной функции библиотеки // Библиосфера. 2018. № 1. С. 35–41

29. Осипов А. А., Дерябкин В. П. Разработка автоматизированной информационной системы библиотеки нормативной документации // Молодой ученый. — 2015. — №6. — С. 52-55.

30. Севрюкова, А. А. Автоматизированные библиотечные системы / А. А. Севрюкова. — Текст: непосредственный // Молодой ученый. — 2015. — № 13 (93). — С. 857-859

31. Ступкин В. В. Методология оценки качества интегрированных библиотечно-информационных систем. Москва: Литера, 2018. 75 с

32. Шрайберг Я.Л., Воройский Ф.С. Автоматизированные библиотечно–информационные системы России: состояние, выбор, внедрение, развитие. – М.: Либерея: ГПНТБ России, 2016. – 273 с.

33. Шорин О.Н. Методы и алгоритмы интеграции большого объема библиографических записей в открытое семантическое пространство: дис. канд. техн. наук / Шорин Олег Николаевич. – Санкт-Петербург, 2016. – 183 с.

34. «Система электронной поддержки образовательных курсов». Курс: Базы данных [Електронне джерело]. – Режим доступу до ресурсу: (https://moodle.pspu.ru/course/view.php?id=880).

35. «Система электронной поддержки образовательных курсов». Курс: Проектирование информационных систем [Електронне джерело]. – Режим доступу до ресурсу: (https://moodle.pspu.ru/course/view.php?id=596).

36. «Система электронной поддержки образовательных курсов». Курс: Разработка приложения по технологии «клиент-сервер» [Електронне джерело]. – Режим доступу до ресурсу: (https://moodle.pspu.ru/course/view.php?id=879).

37. «Методы оценки эффективности ИС» [Електронне джерело]. – Режим доступу до ресурсу: (http://wiki.mvtom.ru/index.php/).

38. Медянкина И. П. Метод анализа иерархий как инструмент выбора электронно-библиотечной системы / И. П. Медянкина Л. К. Бобров // Науч. и техн. б-ки. – 2015. – № 4. – С. 5–14. Шифр НББ: ЗОК1447.

39. Thomas A. Mueck. Index Data Structures in Object-Oriented Databases / Thomas A. Mueck – Luxembourg : Springer Science & Business Media, 2014. – 125 p.

40. Официальный сайт компании Intersystems [Електронне джерело]. – Режим доступу до ресурсу: http://www.intersystems.com/.

41. PostgreSQL. Tutorial inheritance [Електронне джерело]. – Режим доступу до ресурсу: https://www.postgresql.org/docs/8.4/static/tutorial-inheritance.html.

42. S. K. Singh, Database Systems: Concepts, Design and Applications / S. K. Singh – London : Dorling Kinderslay. – 2011

43. Architecture In Object Oriented databases [Електронне джерело]. – Режим доступу до ресурсу: http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.

44. Thomas H. Cormen. Introduction to algorithms. Second Edition / Thomas H. – Cambridge : The MIT Press. – 2005, pp. 521-525.

45. Nilsen P. SQL Server 2005. Библия пользователя/Диалектика 2008. - 1228 с.

46. Дроздова В.И., Крахоткина Е.В., Федоров С.О. Базы данных. Методические указания к лабораторным работам для студентов специальности 351400. Ставрополь, СевКавГТИ, 2015 – 521 с.

47. Каратыгин С.А., Тихонов А.Ф., Тихонова Л.Н. Visual FoxPro 6.0 // М.: Бином, 1999 - 784 с.

48. Хансен Г., Хансен Д. Базы данных. Разработка и управление / М.: Бином, 1999 - 704 с.

49. Баженова И.Ю. Visual Fox Pro 5.0//М.: Диалог МИФИ, 1997 - 320 с.

50. Глушаков С.В., Ломотько Д.В. Базы данных. Учебный курс // Харьков: Фолио; Ростов н/Д: Феникс; Киев: Абрис, 2000. - 504 с.

51. Нильсен Пол, «Microsoft SQL Server 2005. Библия пользователя», 2007 – 351 с.

52. Роберт Виейра, «Программирование баз данных MS SQL Server 2005. Базовый курс», 2007 – 321 с.

53. Пол Уилтон, Джон Колби, «Введение в SQL», 2006 – 554 с.